User Interface for Computer-aided Product Conceptual Design

by

Lan Kong

A Thesis
in
The Department
of
Electrical & Computer Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Applied Science (Electrical Engineering) at
Concordia University
Montreal, Quebec, Canada
Oct 2005

Your file Votre référence
ISBN: 0-494-14267-7
Our file Notre référence
ISBN: 0-494-14267-7

Abstract

Sketching is widely considered an essential and important activity in conceptual design. However, most designers still prefer to use pen and paper as tools as their design tools, although many computer-aided design tools are available in the market. Recently, a great deal of research has been undertaken on sketching and conceptual design to support the development of a new kind of computer-aided design tools that allow designers to sketch freely in the computer system. A major objective of this research is to develop a natural sketching interface for designers in conceptual design.

Based on the literature review on the sketch-based cognitive process, solutions are developed in this thesis to address two major problems that prevent designers from using computer-aided design tools. *First,* because the lack of a detailed description of a designer's cognitive model, designers have never been put in the centre of consideration in the development of a CAD tool. This thesis proposes a new approach for the exploration of the designer's cognitive model by applying the axiomatic theory of design modeling. This new approach is built up on the observation that designers must create a mental model of the user who will use the product to be designed. The user's mental model of a product is the user's anticipation of how the product will work. A designer's cognitive model is used to describe the process of building a user mental model. *Second*, the user mental model underlying the existing CAD tools do not match the actual designer's mental model. To eliminate the gap between two mental models, this thesis develops a prototype of natural interface based on PUI (Perceptual User Interface) theory. This natural interface uses gesture commands to

replace menu and button. ANN (Artificial Neural Network) is adopted to recognize the designer's gesture commands.

Future work of this research includes the integration of this natural sketching interface into a computer-aided design system that supports the activities throughout the conceptual design process.

Thesis Supervisors: Yong Zeng, Yinzhi Lin

Acknowledgements

I would like to thank my supervisor Dr. Yong Zeng for helpful discussions, ideas, and guidance throughout the duration of this work. His enthusiasm, dedication, and expertise were immersed into any little progress of this study. For all of these, I will always be grateful in my lifetime.

I also want to extend my thanks to another supervisor Dr. YinZhi Lin for her help.

I would like to thank all the members of the Design Group in CIISE. Their friendship, support, and suggestions were invaluable to this work.

Finally, I would like to thank my friends and family for their encouragement and support.

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Motivation

People have used sketches as a means of design for years. Sketching plays an extremely important role throughout the entire design process. Designers especially need pencil and paper to release their thoughts so that they may be able to focus on creative design in the conceptual design process. Mechanical engineers use sketches to explore the visual components of a design and to expose the functional requirements. Architects sketch to explore spatial relationships between building components. User interface designers use drawings to describe an interface of an application. Software engineers use sketches to model the logical relationship between objects.

Designers use a wide range of media for sketching: paper, pen, and whiteboards. An important common characteristic of these media is that all of them provide a direct, rapid, and easy way for designers to express their ideas. However, the disadvantages of these media are also obvious. First, because of the widespread use of computers in product design the effort needed to complete a task in early design may be doubled. The designers have to draw a rough sketch on the paper first and then repeat the same task on computer. Secondly, it is difficult to modify a sketch with pen and paper. Even if the draft is not accurate, too many modifications may confuse the designer. Thirdly, designers tend to draw their ideas on papers or notebooks. After several weeks or months, they may not remember where these papers are or what these sketches mean. Finally, it is difficult to store all papers, because paper lacks permanency.

Therefore, it would be beneficial to develop a computer-aided sketching tool to support a designer's conceptual design process. This sketching tool would be able to assist designers to express their ideas freely in a sketching interface and then to transform the completed conceptual design sketches into the existing CAD system. This thesis focuses on the first task, the development of a sketching interface for conceptual design.

1.2 Significance of Sketches

Sketching is an important component in the conceptual design. Most creative fields consider sketching an essential part of the design process. Many designers begin their designs by drawing their rough ideas before moving to the detail design on the computer. Because sketching provides a quick and easy way to represent rough ideas, it is well suited to ill-structured problem solving (Goel 1995).

Cognitive scientists suggest that sketching may allow designers to store their short-term memory quickly into a permanent space. Sketches can serve as an external memory to overcome the limitation of the designer's cognitive abilities (Goldschmidt 1991). Designers cannot store in their minds the huge amount of information contained in a detailed design.

Sketches allow designers to describe the overall concept and then refine or explore the details later.

An important rule proposed by Goel (Goel 1995) is that two types of operations occur in the successive sketches: *lateral* and *vertical* transformations of sketches. A lateral transformation is that a movement is from one idea to a slightly different idea whereas a vertical transformation is that a movement is from one idea to a more detailed and exact version of the same idea. At the conceptual design stage, lateral transformations are the most important. They represent different concepts at an abstract level. By contrast, vertical transformations are required to refine a conceptual design to a detailed design. Experiments strongly suggest that sketches serve the fundamental function of helping designers to alternate between lateral and vertical transformations in the conceptual design stage.

Another important rule in sketching is that sketches allow group collaboration for a design team (Plimmer and Apperley 2002). Current design projects generally require a group to work together to share member's conceptual ideas. A sketch is the best medium to allow visual dialogue between designers. Participants have found that sketches can encourage critical evaluation and discussion (Goel 1995).

1.3 Benefits of Sketching System

It is true that computer-aided design tools are not popular in conceptual design. Most designers consider that sketching with pen and paper is simple, quick, and easy whereas designers have to be trained to sketch with a computer-aided design tool. Furthermore, designers must repeat their sketches on computers by selecting the menu or a widget from the tool bar. These interferences in the design process force designers to be distracted from the most important purpose of the conceptual design, creativity.

On the other hand, a CAD (computer-aided design) system does indeed have many advantages over the use of pen and paper. These advantages include the following: the easy editing of created sketches, less room necessary for keeping the document, potential efficiency of access to the computed design, etc. These advantages motivate many researchers to develop new computer-aided design tools for conceptual designers.

An ideal sketching system should be able to support all the designer's tasks during the conceptual design stage. The conceptual design stage is a complex design stage encompassing the following tasks: collecting user requirements, analyzing requirements, and generating design concepts. Since sketches play an important role throughout the entire conceptual design stage, a sketching system should build a strong relation between sketches and user requirements.

1.4 Objective

Design science and HCI (human-centered interaction) theory provide a solid foundation for the development of a proper sketching system for designers. The present thesis aims to explore the designer's cognitive model in order to reveal how designers accomplish their work. It then indicates the major obstacle facing the current interface of CAD from the HCI viewpoint. Based on the analysis, a natural interface for a mechanical conceptual design process is developed. Unlike other existing sketching systems, this sketching system provides a pen-paper feel to the designer without any complex interference during the sketching process. Our major goal is to create a system that allows designers to sketch in a fast, direct, and natural manner. Our future objective is to integrate this sketching system with another developing system that strongly supports designers in a range of tasks from requirements collection to the creation of proper product concepts.

1.5 Challenges in Sketching System

Though current CAD systems provide many powerful functions, designers reject them because such systems force users to input sketches with a series of structured procedures. In other words, designers have to pick the corresponding symbol from a list and then adjust it and put it in a desired placement. There is an unfortunate gap between existing CAD systems and the designer's traditional way of dealing with these tasks. Based on HCI theory, this gap is due to the user's mental models built on the previous system.

An important difference between the conclusions drawn in this thesis and the results produced in other studies about the sketching system is that most other researchers have focused on designing computer tools but have paid less attention to the designers themselves. The following figure shows this difference.

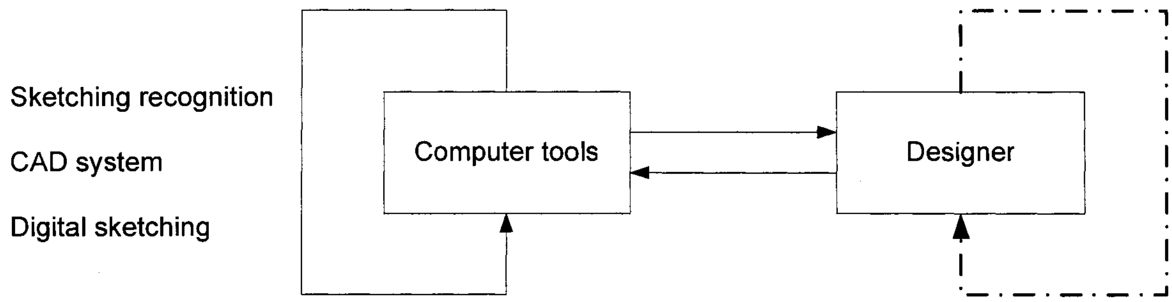


Figure 1 Sketching System Architecture

Many sketching systems adopt different technologies to support designers, such as sketch recognition just as Figure 1 shows. However, these sketching systems cannot well satisfy designers in a real world. An important unrevealed cause behind this is that UI designers cannot build a proper cognitive model for the users of a sketching system, such as mechanical designers. A cognitive model of users generally enables UI designers to design the proper user interface of a computer system. The difficulty in building a mechanical designer's cognitive model is that people always consider this model to be an informal model that cannot be described by formal explanation. Without a formal cognitive model, it would be difficult to solve the conflict between the informal cognitive model and a formal computer sketching system. One of this thesis's emphases is to apply the axiomatic theory of design modeling (Zeng 2002) to propose a formal representation of a designer's cognitive model.

In order to develop a proper system to support designers in the conceptual design stage, details describing how designers work during the conceptual design stage should be gathered. Unfortunately, it is very difficult to understand the thinking process of human beings. Although, researchers in cognitive science have endeavored to reveal the secret of design for a long time, the overall pattern of design thinking has not been discovered. Furthermore, the designer's cognitive model is significantly different from general user cognitive model. Many existing cognitive models that are used widely and successfully in HCI, such as GOMS (Goals, Operators, Methods, and Selection Rules), cannot be applied to the conceptual design stage. These models fail to explain how to design because the logical steps used in the design process are totally different from those used generally. Therefore, this thesis explores another approach in the attempt to build a designer's cognitive model based on the design theories.

1.6 Contributions

Two major contributions to the creation of sketching interface are presented in this thesis:

1) Based on the axiomatic theory of design modeling, a designer's cognitive model is described. A series of characteristics and fundamental functions of sketching are derived from this model.

This designers' cognitive model can guide software designers in their understanding of the design process. The specific design process requires that software designers alter their traditional user interface so that it is more satisfactory.

2) Based on the analysis of the cognitive model of designers, a natural interface developed from perceptual user interface (PUI) is proposed and developed. Artificial neural network is adopted to recognize users' gestures.

PUI is a new concept of HCI that is found from more and more real world applications. In conjunction with developing computer technologies, the PUI allows users to exceed the limitations and constraints of the traditional computer user interface. Applying it to the development of a proper interface for designers working in conceptual design would be an additional successful instance.

1.7 Structure of this Thesis

The rest of the thesis is organized as follows. Chapter 2 reviews the current studies of design science and introduces existing sketching systems. Chapter 3 presents a brief introduction to the axiomatic theory of design modeling. Chapter 4 describes the details of a designer's cognitive model. This cognitive model guides UI designers to develop a proper interface for a sketching system. Chapter 5 presents a prototype of a sketching system based on the analysis of a designer's cognitive model. This natural interface is inherited from PUI. Finally, Chapter 6 and 7 conclude this thesis and lists some future research directions.

Chapter 2

Review

This section attempts to collect books and papers from journals, conferences on cognitive science, psychology, design science and HCI and summarize other researchers' achievements. The topic of this thesis has two key words: *Design* and *UI*. Cognitive science, and design science focus on how products are designed and improved whereas psychology and HCI emphasize how people recognize the world and how users can be helped to operate a new system. In the first two domains, researchers explore the design process and the activities of designers and then summarize the characteristics of design. The current results arising from these domains fully or particularly deal with the following questions:

- *What is a design problem?*
- *How to solve a design problem?*
- *What are the characteristics of a design process?*

The next section will review these results. Furthermore, with the ongoing development of design science, several design theories have arisen in the last decade. Because this thesis is based on one of these design theories, the axiomatic theory of design modeling is introduced in the same section. The second section summarizes the current results of psychology and HCI to indicate how to design a good user interface of a computer system. Synchronously, a new explanation of the mental model based on the axiomatic theory of design modeling is introduced.

2.1 Design theory

2.1.1 What is Design?

Designing is one of most important characteristics distinguishing human beings from animals. People have been designing for thousands of years beginning with the first tool. In summary, in designing, people rely on their intelligence to create a new product. However, even though human beings have had a long history of designing artifacts; the characteristics of the design process and of human thinking have been couched in uncertainty and obscurity for centuries. However, recently researchers from distinct domains have shown a growing interest in designing process. Researchers from cognitive science, psychology, computer science, and HCI have begun to analyze the designing process, to compare experts and novices, and to summarize the results of their experiments all with the intention of improving the designing capabilities of human beings.

In addressing some of the characteristics of designing, Cross has summarized what other researchers and expert designers' have said about design (Cross 1999). These characteristics of design are listed briefly as follows:

1. *Design is rhetorical.* This characteristic cannot be easily understood by people. In another words, it means that design is persuasive. It indicates that customers may not be able to recognize the particular values of one new product until designers can prove this product to be particularly suited for specific proposes. This characteristic also

suggests that designers may create a new product that does not correspond to their previous goals.

2. *Design is exploratory*. People cannot find a solution by providing all the specifications of one problem. In most cases, designers have to modify one existing solution to match the desired specifications or they have to create a totally inexistent concept to satisfy the requirements. The activities of designers are like explorations in an unknown territory.

3. *Design is emergent*. What this means is that those relevant features emerge in putative solution concepts. This characteristic of design implies that a design problem is an ill-defined problem that requires the designers to develop the problem and the solution together. This kind of problem cannot be solved simply by collecting and synthesing information. The next section tackles the complexities of this characteristic.

4. *Design is opportunistic*. A designer, even an expert designer in a given domain, cannot predict which direction should be taken at the beginning of the design. This characteristic implies that design is an exploration of an unknown field. However, unfortunately, it is difficult for designers to realize which direction should be taken.

5. *Design is abductive*. Cross proposes that abduction is the kind of reasoning adopted by designers to create a new concept. He considers that abduction is different from induction and deduction and it is the necessary logic of design. However, some researchers propose a new pattern of logic for designers currently (Zeng and Cheng

1991; Eekels 2000). This new pattern is explained in more detail in the following section and it is also one of the points emphasized in this thesis.

6. *Design is reflective.* It reflects the typical characteristic of the thinking process of designers. Romer points out that designing contains both complex internal representations (thinking, evaluation, decision) and external representations (writing, drawing, speaking) (Romer, Leinert et al. 2000). There is a dialogue or conversation between the internal and external representations. Designers rely on the external media, such as sketches, to release their mental load and create half-formed ideas.

7. *Design is ambiguous.* Designers may create rough solutions, but also keep them in mind as long as possible. It is necessary to keep imprecise and often inconclusive solution concepts.

8. *Design is risky.* One must acknowledge that even famous designers make their reputations by taking risks.

The following sections present more detail about these characteristics of design based on the current research contributions.

2.1.2 Design Theory

In the engineering domain, two strategies are widely used: bottom-up and top-down strategies. The bottom-up strategy is to construct design theories by generalizing design activities while the top-down strategy is to establish design theories that are derived from axioms (Zeng 2002). A design theory addresses the nature and models of the design process, of design objects, and of design knowledge. It can be applied to improve the design process.

Eekels addresses that researchers studying in design methodology not only aim at understanding the phenomena of design, but also use their understanding in order to change the way the design process is carried out (Eekels 2000).

Researchers following the bottom-up strategy observe the designer's activities and then base general design theories on those activities. A series of methods can be used to achieve this goal. Some of them are speculation, retrospection, and protocol analysis. Several important design theories are included in this field: the systematic design methodology (Hubka 1984; Hubka and Eder 1988; Pahl and Beitz 1988), the theory of inventive problem-solving (Altshuller 1984), Axiomatic design (Suh 1990), the decision-based design theory (Allan and Mistree 1997), and the artificial intelligence-based design (Gero 2000).

On the other hand, some design theories are derived from axioms directly. These axioms are self-evident truths, on which these theories are based. Properties about design activities can be derived from these axioms by using patterns of logic. The major theories contained in this branch are the following the general design theory (Yoshikawa 1981; Tomiyama and Yoshikawa 1987), the axiomatic theory of design information (Salustri and Venter 1992), the formal theory of design (Braha and Maimon 1998), and the axiomatic theory of design modeling (Zeng 2002). This thesis focuses on the last one, the axiomatic theory of design modeling, to explore different aspects of design activities and to develop a computer-aided tool to support mechanical designers.

2.1.3 Design Problem

In the problem-solving literature, there are many descriptions of the results concerning design problems. While people investigate puzzle-game domains, most researchers share one common intuition, namely, that there are important distinctions between design problems and puzzle problems (Goel 1992) Playing chess is significantly different from designing a machine. Currently, most researchers in cognitive science recognize this difference and classify the problems into two categories: ill-structured problems and well-structured problems. Design problems belong among the ill-structured problems whereas cryptarithmetic problems belong among the well-structured problems. The distinctiveness of design problems makes the design process more complex than general problem solving. Goel lists several differences between design problems and well-structured problems (Goel 1992).

- The constraints of problems. In a puzzle game, the constraints are logical or constitutive of the task. However, the constraints of a design problem are significantly different. Some of them are nomological; some of them are economic; some of them are social. And a common and important characteristic of these constraints is that they are not definitional or constitutive of the task. In other words, they are negotiable.
- The size and complexity of the problem. Generally, people can solve a puzzle problem in hours or days. However, designers have to spend weeks even months, to find a solution to a specific design problem.

- The roles to decompose problems. People should solve both problems by decomposing them into sub problems. However, a puzzle problem can be decomposed by the logical structure of the problem. On the contrary, designers decompose the design problem by the physical structure of the world, practice within the community, and personal preference.
- The answer of problem. A significant difference between ill-structured problem and well-structured problem is that there is no right or wrong answers for an ill-structured problem while in most cases of well-structured problem people can easily recognize the right or wrong answers.

One important characteristic of design problem that many designers observe in different fields is that design problem cannot be formulated before they have been solved (Zeng and Cheng 1991). When Ritte talked about design problem, he said that 'you cannot understand the problem without having a concept of the solution in mind; and you cannot gather information meaningfully unless you have understood the problem but you cannot understand the problem without information about it' (Rittel 1972). Because the explanation of this critical characteristic of design problem relates the design process and design solution, the following section explores the detail of this characteristic.

2.1.4 Design Process

The design process is generally described as a complex and fastidious mental activity. The purpose of design is not only analyze the given constrains of a desired problem and seeking an existing solution for this problem, but also requiring designers to create a new approach or build a non-existent product. The need to define and understand design process has become more focused in recent years. This has been because of various attempts to realize the benefits of CAD to the design process. The design process varies from product to product and industry to industry. Nonetheless, a generic diagram of the activities that must be accomplished for all products can be shown in Fig 1 (Ullman 2002).

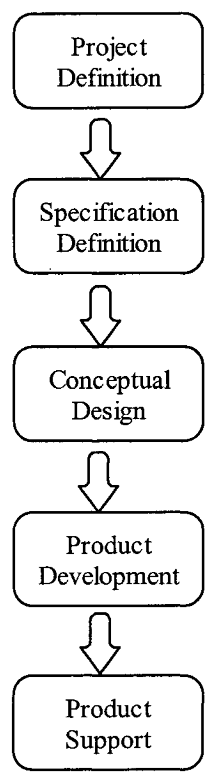


Figure 2 Product Design Process (Ullman 2002)

Figure 2 describes the design process in an industry domain. There is a more abstract description about design process in design thinking. Studies reveal that 'analysis – synthesis – evaluation' is a design method or process adopted and discussed in the literature(Ho 2001). Tovery addresses a similar viewpoint of design process (Tovey 1997). A typical design process is listed as following.

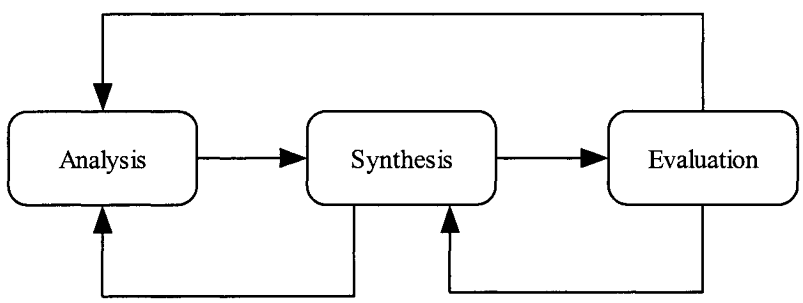


Figure 3 Typical Design Process (Tovery 1997)

For an individual designer the design process seems like that: start thinking about the problem; propose a solution of the problem; then verify this solution; if the solution can satisfy the specific problem, then stop and implement this solution; if this solution has some shortages, then refine this solution. Exactly, the both above description of design process reflects each step of design thinking of individual designer. "Analysis" means to decompose the big problem into sub-problems; "synthesis" refers to the recomposition of sub-problems in distinct ways; 'evaluation' indicates that to verify the new solution and get feedback of this new solution. One example in architectural design domain is mentioned by Rowe (Rowe 1987). Designers try to solve the design problem, an ill-structured problem, and then divide the original problem into several different sub problems until the sub problems can be dealt with by designers. After designers have a series of sub solutions for each sub-problem, they can synthesize these solutions together.

Another popular description of design process is given by Pahl and Beitz (Pahl and Beitz 1997). The phases of the design process are differentiated as shown in Figure 4.

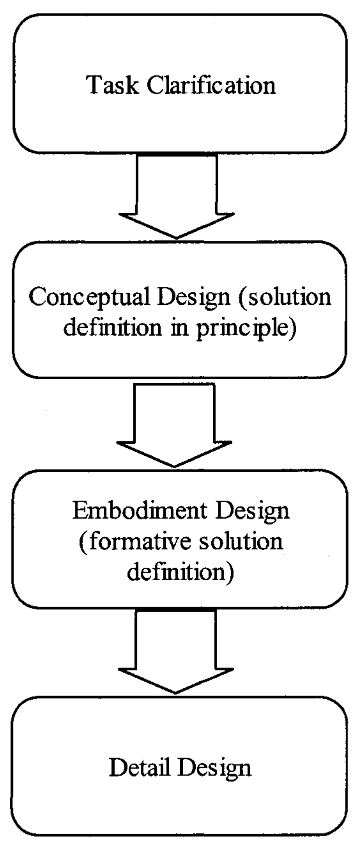


Figure 4 Pahl and Beitz's Design Process (Pahl and Beitz 1997)

Because an experience-based approach to handling problems is not possible, the design process is seen as a problem-solving process by most researchers (Romer, Leinert et al. 2000).

2.2 Sketching in Design

Section 2.1 suggests that the general process of design always includes a conceptual stage. In conceptual design, a series of different types of drawings are used. They can be called sketches. Researchers believe that sketches have a strong relationship to the designer's innovation and creativity. Researchers have observed the activities of designers and analyzed the characteristics of their sketches. The present section of this thesis aims to review the results in current research fields to reveal the role of sketches in the conceptual design process. A clear understanding of the relationship between cognition, sketches, and sketching would establish an important premise for the creation of a computer tool for designers.

2.2.1 Significance of Sketching in Design

There has been a long tradition of using sketches as part of the design process. Because of the unanticipated character of design, individual designers usually have pen and paper available to support the breakthrough in their thinking (Verstijnen, van Leeuwen et al. 1998). Designers can use sketches to express their rough ideas, such as describing the functions of the new product. Because designers' thoughts generally arise suddenly in their minds, designers think it is essential to express and catch them immediately and without frustration.

That explains why it is that sketches can sometimes be found on the backs of envelops, on newspapers, or scraps of paper. In contrast to presentation-sketches, researchers call the sketches created in the conceptual design stage "idea-sketches". Idea-sketches can be more abstract and unstructured compared to presentation-sketches.

In a series of experiments, Kavakli investigated designers' thinking processes using sketches. She noted that designers do not require external aids, such as sketches, when they undertake combination and recombination activities. However, decomposition of parts and analysis the relations of a structure and recomposition a new structure with new parts and relations are more difficult to designers to do internally. In this case, external aids, such as sketches or diagrams, can help designers to make a more detailed analysis (Kavakli, Scrivener et al. 1998). Designers can store information externally; in that way, the processes can be reduced. Gross et al. studied the design examples in the architectural domain and argued that sketches play an important role in the design process because they embody abstract and high level design ideas (Gross, Ervin et al. 1988). A sketch can provide a graphic means which contains the related conceptual knowledge and helps designers to recall and manipulate the visual representations.

2.2.2 Roles of Sketching in Design

The use of sketches is an important part of the processes of designing. An engineering designer has said, "I draw everything even if it is potty". It is obvious to understand how the processes between sketching and thinking work would give us deeper insights of the design process. Hsien-hui and Gero have said that sketches made by designers play different roles for both designers and the design process (Tang and Gero 2001). Furthermore, Cross (Cross 1999) has suggested that sketching is fundamental, creating a kind of "dialogue" for the designer to reflect the different stages of the design process.

However, why do designers try to draw at all? One obvious reason is that one of the final requirements of design is usually a drawing or a series of drawings, which describe a model of the new product. In the architectural and mechanical domains, it is usual to produce a model that clearly reflects the future product.

Sketches in design promote the recognition of emergent features and properties of the solution concept (Cross 1999). Several researchers have similar opinions about the basic role of sketches in design. The "seeing as" of Goldschmidt, the "moves" of Schon and Wiggins, the "lateral transformations" of Goel, and the "focus shifts" of Suwa and Tversky all reveal this phenomenon of the sketches' reinterpretation and emergence (Purcell and Gero 1998). Actually, to solve a specific design problem, there are a lot of possible solutions. Designers have to explore some of them to find a better one. An important function of sketches is to aid designers to shift from one space to another. In summary, sketches help designers move to a new alternative or a new way of reinterpretation of design.

The specific characteristics of sketches, that is, their ambiguity and unstructured characteristics, help designers handle different levels of abstraction. Both Goel and Goldschmidt indicate that sketches have two basic properties, denseness and ambiguity. As we know, a design problem is an ill-structured problem. Therefore, just like Goel argues that ill-structured problems, like the preliminary phase of a design problem, need ill-structured diagrammatic representations (Goel 1992). The two basic characteristics of sketches prove that sketches are the proper representations of design problems.

Sketches help designers identify the details of the solution and recall relevant knowledge. The results of analyzing the design process suggest that designers have to keep the overall concept and the detailed aspects of that concept in mind at same time. However, not all of these detailed aspects of the concept are necessary in the preliminary phase. It is most important for the designer to identify only some of the critical details of this concept and then refine them based on related knowledge, which is stored in the designers' long-term memory. There is a lot of this related information about how to solve the problem. However, only some of it can be useful. At this point, sketches serve as an external memory for detailed inspection. The design process is also a selection phase, during which designers rely on their knowledge to explore possible solutions. Furthermore, Schon and Wiggins compare the results of an expert designer and with those of a novice (Schon and Wiggins 1992). The analysis suggests that experts can access different types and amounts of knowledge more efficiently while exploring potential solutions.

Sketches assist designers in problem structuring by presenting solution attempts. The numbers, texts, and symbols are included in sketches. All of these information aid designers to explore space problems and to find corresponding space solutions. Sketches enable designers to describe the characteristics of the problems while expressing the possible solutions. To designers, their sketches in the preliminary phase can provide visual cues for the association of functional issues.

2.2.3 Types of Sketching in Design

Generally, there are three kinds of sketches in engineering. Ferguson analyzes the sketches in the preliminary design phase and then identifies the following three kinds of sketches(Ferguson 1992):

- *The thinking sketch*: engineers used to focus and guide nonverbal thinking.
- *The prescriptive sketch*: used by an engineer to direct a draftsman in making a finished drawing.
- *The talking sketch*: produced during exchanges between technical people in order to clarify complex and possibly confusing parts of the drawing.

Sketches can also be classified into five levels based on their complexity level. Tovey and Porter list the following five levels of sketches (Tovey, Porter et al. 2003).

1) Complexity level one: monochrome line drawing, no shading or colour, uniform line thickness.

2) Complexity level two: monochrome line drawing, no shading or color. Line thicknesses vary to give emphasis. May include brief annotation.

3) Complexity level three: monochrome with rough shading to suggest form. May be annotated.

4) Complexity level four: line and shading, may include color and graduation.

5) Complexity level five: color illustration to show what the product looks like. Color, shading, shadows, annotations, dimensions.

2.3 Computer-aided Sketching System

Researchers in the 90's developed several computer-aided system that underlie recent achievements in several areas: artificial intelligence, cognitive science, design science, and human-centered interaction. All of these computer-aided systems can be divided into three categories: sketch digitization, sketch regonition, and 3D sketching system (Buchal 2002).

2.3.1 Sketch Digitization

The first solution is to imitate pen and paper to develop freehand sketching systems and keep the original sketches without any processing. The advantage of these systems is to retain all the features of the traditional pen-paper technique with the additional benefits of computer-aided systems, such as editing, undoing, and storage.

2.3.2 Sketch Recognition

Such systems try to recognize the sketch input and then to replace the rough sketch with regular objects. The difficult challenge in these systems is to recognize the freehand sketches. Most such systems try to analyze the geometric relationships between shapes and to infer the final objects. The algorithms of such systems were developed many years ago. Two typical systems are SILK and Cocktail Napkin.

SILK (sketching interfaces like krazy) is a sketching system providing an interactive user-interface design tool for UI researchers (Landay and Myers 2001).

Researchers at the University of California, Berkely and Carnegie Mellon University have designed, implemented, and evaluated this informal sketching tool that combines many of the benefits of paper-based sketching (Landay and Myers 2001). UI designers can quickly draw an interface with SILK, which SILK then has the ability to recognize. One benefit of SILK, which is unlike other paper-based sketching, is that designers can simulate their sketch interface with SILK.

When designers draw an interface using a set of components and gestures, SILK can recognize them and suggest to designers what these units are. SILK can recognize four primitive components: rectangle, squiggly line, straight line, and ellipse. All of these components are single-stroke shapes, which means that designers must draw them without lifting up until the stroke is finished. These components combine to form widgets. As soon as SILK recognizes the widget, designers can switch to SILK's run mode to show what this widget does. After designers are satisfied with the sketch, SILK may create a new window that contains real widgets corresponding to the sketch.

The usability test showed that SILK can effectively support the early stages of UI design. Even if there are some problems in the widget recognition, most designers admitted that SILK is a simple but promising digital tool.

The Electronic Cocktail Napkin is a computational environment for working with design diagrams developed by Gross (Gross 1996).

Gross argues that because diagrams are quick and easy to make, the designer can rapidly explore a variety of solution types without the effort and commitment of making more careful drawings.

Gross observes that there are two obstacles working against thinking quickly and abstractly with today's CAD software: precise and structured internal CAD representations and tedious mouse based human-computer interfaces. He indicates an approach in his report: firstly, adopt a paper-like, or pen-based, interface that allows designers to draw directly what they have in mind, with varying degrees of precision, ambiguity and abstraction. Second, provide internal representations that can tolerate ambiguity and incompleteness, yet which can be made more formal and structured as designing proceeds. By providing facilities for recognition, parsing and constraint management, the Cocktail Napkin enables the designer to move gradually from unstructured diagrams toward more formal and structured CAD representations.

He emphasizes that without a representation of the design's structure the computer can do little more than paper and pencil. Therefore a program that supports hand-drawing for design must have capabilities for recognition and parsing. The Cocktail Napkin program adopts a three-step process. Firstly, the program recognizes hand drawing multistroke symbols or glyphs drawn on a tablet. Secondly, it analyzes spatial relations among diagram elements. Thirdly, it matches elements and relations it finds against previously defined configurations, thereby parsing the diagram.

The Cocktail Napkin Drawing Environment can be described as follows: the designer's drawing appears in the work area; a text interaction area at the screen bottom displays messages and allows the designer to type names for new elements; an array of icons at the top shows the pages of a sketchbook of previously made diagrams; trace tabs at the top left enable the selection of various layers of simulated tracing paper, and buttons at the left of the screen provide commands to clear the screen, hide and remove layers. In addition, commonly used operations (pick, erase, clear screen, trace overlay, copy) are provided as gesture commands. Most pen-based drawing environments (e.g. the Apple Newton) rectify shapes and characters immediately upon recognition and discard the raw, as drawn, glyph data. This may be appropriate as designing approaches the schematic phase, but for conceptual design it is better to simply display the input glyphs as drawn. The raw character of hand drawn glyphs reminds the designer of the rough level of thinking instead of reading precision into the designer's intentionally unrefined marks. It also permits glyphs to remain indeterminate or ambiguous.

In summary, the Electronic Cocktail Napkin is a prototype-diagramming environment for conceptual designing based on the premise that tools for early stage designing must not demand great effort, commitment or precision. It has two major components: facilities for recognition and parsing constraint management routines, and more general support for drawing management.

2.3.3 3D Sketching System

The last sketching system can allow designers to draw sketches directly in 3D. The typical example is FSD (Fast Shape Design) system.

FSD (Fast shape design) developed by Casper (van Dijk 1995) is a prototype surface modeler to support industrial engineers during the conceptual design process.

Casper analyzed the conceptual design phase based on the well know Pahl and Beitz model (Pahl and Beitz 1988). The first step was to look for function structures and essential problems; the second step was to find solution principles; next, all of these principles were combined into design concepts; finally these concepts are evaluated.

Casper listed some initial requirements on computer-aided conceptual design.

1) Easy data entry: entry of geometric data must be quick and easy.
2) Hand movements: we need to maintain a direct relation between hand movements and modeling operations.
3) Imprecise data: it would prohibit an unhindered flow of thoughts if the designer had to think about exact sizes and dimensions at the early conceptual design stage.
4) Switch to details: top-down development from a rough sketch model to a well defined and detailed concept is, in general, not the way designers work. We have to allow the designer to zoom in on specific features. The system has to support switching between different levels of detail.
5) Review alternatives: it must be possible to view design alternatives for comparison.
6) Separate conceptual design system: several object representations are needed to support the full design cycle.
7) Clay modeling: in such a system, a shape can be interactively deformed and evaluated.

In summary, FSD is a prototype CAD system that tries to realize the requirements. Using GUI the designer makes a model. The interaction between the FSD and the users is through mouse and buttons. The FSD is most useful in the design of free-formed surfaces, especially in the building of a complex 3D object. However, it is less suited for modeling objects that consist of simple geometric primitives.

Chapter 3

Axiomatic Theory of Design Modeling

This chapter briefly reviews the axiomatic theory of design modeling proposed by Dr. Zeng, which is the foundation of this thesis (Zeng 2002). The axiomatic theory of design modeling is a logical tool for representing and reasoning about object structures. It provides a formal approach that allows for the development of design theories following logical steps based on mathematical concepts and axioms.

3.1 Axioms

An axiomatic system contains a set of primitive concepts and axioms. Axioms are also called postulates. Primitive concepts are usually informally described and left undefined. Axioms are self-evident truths that can be taken as the basis for inference. They often look simple and trivial (Zeng 2002).

Design is an activity in which products are created that can function in a desired manner. Design involves both nature and human thought. Nature is where the products are supposed to function while human thought is where the design ideas are generated. All objects appearing in the design process are called design objects, which include design requirements, design solutions, and design knowledge. These objects reside in nature and human thought as well. The design process deals with the relations between these objects. Hence, the modeling of design depends on the assumptions underlying the nature of design objects and the design thought process. The following two groups of axioms address assumptions for this theory.

Axioms of objects

1) *Axiom 1*. Everything in the universe is an object.

2) *Axiom 2*. There are relations between objects in the universe.

Axioms of human thought

1) *Axiom 3*. Human beings are bounded by rationality.

2) *Axiom 4*. Human beings do not recognize objects accurately.

3) *Axiom 5*. Causal relation is the only plausible relation in all relations between causes and effects.

To present this axiomatic system in a logically consistent way, the following mathematical symbols are used throughout this thesis:

1. Predicate symbols: $\supseteq$ (inclusion), = (identity), $\neq$ (inequality)

2. Operation symbols: $\cup$ (union), $\cap$ (intersection), $\otimes$ (relation), $\oplus$ (structure)

3. Logical symbols: $\neg$ (negation), $\wedge$ (conjunction), $\vee$ (disjunction), $\forall$ □ (universal quantification: read as "for every"), $\exists$ (existential quantification: read as "there exists one"), $\exists!$ (existential quantification: read as "there exists exactly one"), $\rightarrow$ (logical implication), $\leftrightarrow$ (if and only if), $\xleftrightarrow{\Delta}$ (defined by)

4. Auxiliary symbols: (,), [,].

The logical symbols used in the axiomatic theory have the same meaning that they have in other branches of mathematics and logic (Van Dalen, Doets et al. 1978). Auxiliary symbols are self-evident in the context. Predicate and operation symbols will be described or defined based on the axioms of objects.

3.2 Axioms of Objects

Axiom 1. Axiom of the Universal Object

Everything in the universe is an object.

In this axiom, universe and object are two primitive concepts. Informally, universe is the whole body of things and phenomena observed or postulated. An object is any element that can be observed or postulated in the universe. This axiom looks trivial and simplistic. However, it makes this theory different from set theory where concrete and abstract objects are distinguished by set and element. In our theory, the universe is the only abstract concept, which sets the boundary for the discourse of our discussion. Every other object is treated as the same in that it is an object in the universe. This brings convenience into the uniform representation of design objects in the evolutionary design process.

Axiom 2. Axiom of Object Relation

There are relations between objects in the universe. Symbolically,

$$A \sim B, \ \forall A, B, \quad (1)$$

where A and B are objects. A~B is read as "A relates to B". A relation of one object to itself is called the relation on the object itself.

In this axiom, relation is an aspect or quality that connects two or more objects as being or belonging or working together or as being of the same kind. Relation can also be a property that holds between an ordered pair of objects. This axiom has many implications. Obviously, different types of relations will lead to different concrete axiomatic systems. Corollary 1 defines an inclusion relation between two objects:

Corollary 1 Every object in the universe includes other objects. Symbolically,

$$A \supseteq B, \ \forall A \ \exists B, \quad (2)$$

where B is called a subobject of A. The symbol $\supseteq$ is inclusion relation.

Corollary 2 Every object in the universe interacts with other objects. Symbolically,

$$C = A \otimes B, \ \forall A, B \ \exists C, \quad (3)$$

where C is called the interaction of A on B. The symbol $\otimes$ represents interaction relation. Interaction relation is idempotent but not transitive or associative. Based on the Corollaries 1 and 2 above, a new operation can be developed as follows:

Definition 1 Structure operation, denoted by $\oplus$, is defined by the union of an object and the interaction of the object with itself.

$$\oplus O = O \cup (O \otimes O), \forall O, \tag{4}$$

where $\oplus$ O is the structure of object O.

The structure operation provides the aggregation mechanism for representing the object evolution in the design process.

3.3 Axioms of Human Thought

Different from traditional sciences dealing with understanding nature, design studies attempt to investigate both nature and the designer's thought at the same time. By thought we mean the sum of information that human beings can think of and reason with. However, in this thesis, we will not specify what is the carrier of the thought. As such, a new object, world, is defined.

Definition 2 World

The world is an object in the universe, which is made up of two objects: nature and human thought. The world, nature, and the human thought are denoted by W, N and M, respectively, i.e.

$$W = N \cup M. \tag{5}$$

By applying structure operation ⊕ defined in Equation (4) to the object W in Equation (5), the structure of the world, ⊕W, is

$$\begin{aligned} \oplus W &= \oplus (N \cup M) \\ &= (N \cup (N \otimes N)) \cup (M \cup (M \otimes M)) \cup (N \otimes M) \cup (M \otimes N) \\ &= (\oplus N) \cup (\oplus M) \cup (N \otimes M) \cup (M \otimes N) \end{aligned} \tag{6}$$

Corresponding to Equation (6), we can define four objects: the natural system, the human rational system, recognition, and action.

Definition 3 Natural system

The natural system is the structure of nature. Symbolically,

$$\oplus N = N \cup (N \otimes N) \tag{7}$$

Natural law L is a subobject of the relation N⊗N, and

$$L \subseteq N \otimes N \tag{8}$$

Definition 4 Human rational system

The human rational system is the structure of human thought. Symbolically,

$$\oplus M = M \cup (M \otimes M) \tag{9}$$

Knowledge K is a subobject of the relation M ⊗ M, and

$$K \subseteq M \otimes M \tag{10}$$

Definition 5 Recognition

Recognition is a relation f^R from nature N to human thought M.

$$f^R \subseteq N \otimes M \tag{11}$$

Definition 6 Action

Action is a relation f^A of human thought M to nature N, which influences nature N by implementing a plan or a design produced by human thought M. Symbolically,

$$f^A \subseteq M \otimes N \tag{12}$$

Design is an activity happening in the human rational system. The results have to be sent back to the natural system through action. Once a design has materialized in nature, it has to follow natural laws. Whether and how a design can survive in nature depends on the answers to the following questions:

1. What is the character of nature?
2. What is the character of recognition?
3. What is the character of the human rational system?

The answer to the first question can be philosophical and theological, for it depends on how human beings understand the world. Any answer to this question would only be relatively true with respect to the answers to the last two questions, since it is itself a result of human recognition and reasoning (Knight 1999). In accordance with human commonsense, three axioms are developed to address the last two questions.

First, according to Corollary 2, the structure of an object depends on what objects are included in the object. As we can see in Equation (6) and Definition 5, the goal of the recognition process is to define the structure of an object in nature N with the object(s) in human thought M. If the human thought has a one-to-one correspondence to nature, then they would have perfect knowledge. This is the axiom of correspondence in Yoshikawa's General Design Theory (Yoshikawa 1981). But it is against human commonsense. The axioms of bounded rationality and recognition address this characteristic.

Axiom 3. Axiom of Bounded Rationality

Human beings are bounded by rationality.

This axiom is adopted from Simon (Simon 1969). In this axiom, rationality is the quality or state of being that has reason or understanding. The main manifestation of this axiom is the limitation of resources (e.g., time and memory) in the rational system. A direct result is the limited number of objects in the human thought M. That is

$$M = \bigcup_{i=1}^{n} O_i^a \tag{13}$$

where i and n are finite natural numbers. Each object O_i^a is called a primitive object.

Substituting Equation (13) into Equation (10),

$$K \subseteq (\bigcup_{i=1}^{n} O_i^a) \otimes (\bigcup_{i=1}^{n} O_i^a) \tag{14}$$

Since n is a finite natural number, knowledge K is limited. This means that the amount of human knowledge is limited.

The choosing of primitive objects is artificial. It can have observable and/or measurable properties, or even objects with complex structure. One of the tasks of scientific research is to look for the minimum number of primitive objects to describe various natural phenomena.

Human Recognition Process

Another issue in the human recognition process is the nature of recognition. This is addressed by the following axiom:

Axiom 4. Axiom of Recognition

Human beings do not recognize objects accurately. That is,

$$(O' = f^R(O)) \wedge (O' \neq \bar{O}), \forall O \subseteq N \exists O' \subseteq M \quad (15)$$

Definition 10 Causal relation

The causal relation is the relation from cause to effect in the human rational system.

Axiom 5 Axiom of Causality

The causal relation is the only plausible relation in all relations between causes and effects.

$$e_e = C^r(e_c), \ \forall e_c \subseteq O^c \subseteq M, \exists!\, e_e \subseteq O^e \subseteq M, C^r \subseteq O^c \otimes O^e \quad (16)$$

where C^r is a causal relation between causes O^c and effects O^e. This axiom determines that human reasoning is not symmetric. Only deductive reasoning is deterministic. All other reasoning modes are implausible.

Chapter 4

Designer's Cognitive Model

In building a sketching system for conceptual design, two fundamental parts must be studied: designers and computer tools as shown in Figure 5.

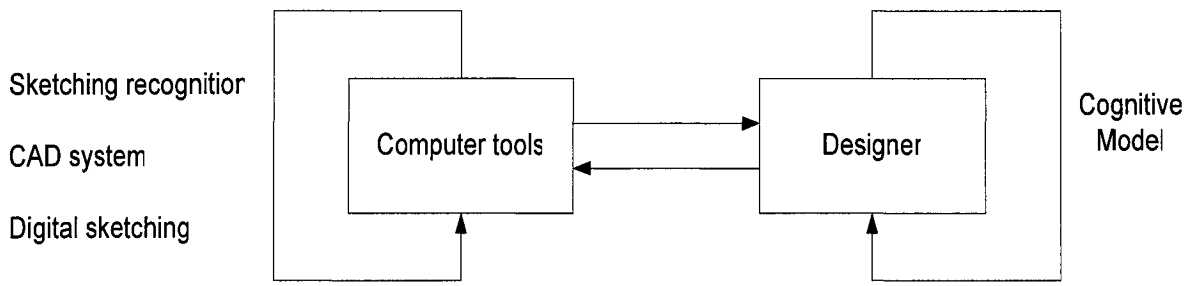


Figure 5 Sketching System Architecture

A good interface for a conceptual designer depends on a well-formulated designer's cognitive model and robust technologies to support a designer's conceptual design activity. Many researchers propose distinct interfaces for designers in the conceptual design stage. However, these approaches generally focus on the computer tools part in the sketching system architecture. Only a few articles discuss how to develop an appropriate interface based on a designer's cognitive model, which is a fundamental part guiding the UI designer to develop a proper interface. The difficulty in building a designer's cognitive model lies in that the current general models cannot accurately describe designer's activities. This chapter proposes a new approach to exploring a designer's cognitive model using the axiomatic theory of design modeling based on current results from a mental model.

4.1 Cognitive Model

A cognitive model may include a "circle & arrow theory" of how some aspect of cognition is structured (e.g. information processing stages), or a set of equations with the proper input-output specifications and some internal structure that is believed to represent some aspect of cognition (Vijay 2005). People can examine whether the model can foresee any of traits of the aspect of the cognition that it claims to govern.

There are three main ways that cognitive models are successfully adopted in HCI. The first way is by using cognitive models to examine the efficacy of different designs. One of these successful theories is the GOMS family of techniques. This theory aids UI designers to create and choose better interface designs. The second way is by using cognitive models to provide assistance such as in the form of an embedded assistant. These models help users to modify interactions in order to improve efficiency. Cognitive tutors are the typical examples employing this theory. The third way is by using cognitive models to substitute for users. These models are used to populate synthetic environments. A good example is the simulation of fighter aircraft crews in a simulated war scenario (John 1998).

Unfortunately, none of these methods can be applied to a designer's cognitive model. These methods are based on rational theory that suggests how users make their decisions. However, the current result of cognitive science reveals that the logic adopted in design is not of the normal three kinds of logic but is a type of logic called recursive logic (Zeng and Cheng 1991; Zeng 2002). It is necessary to explore what a designer's cognitive model is.

4.1.1 Mental Model

Before a proper cognitive model for a designer is introduced, an important concept related to the mental model of the human mind will be reviewed. Mental models are images of reality. They are used to understand specific phenomena. Norman describes them as follows: "In interacting with the environment, with others, and with the artifacts of technology, people form internal, mental models of themselves and of the things with which they are interacting. These models provide predictive and explanatory power for understanding the interaction." (Norman 1988)

Some researchers summarize the characteristics of mental models as follow (Gentner and Stevens 1983):

1. They are incomplete and constantly evolving,
2. They are usually not accurate representations of a phenomenon; they typically contain errors and contradictions,
3. They are parsimonious and provide simplified explanations of complex phenomena,

4. They often contain measures of uncertainty about their validity that allow them to be used even if incorrect, and

5. They can be represented by sets of condition-action rules.

These characteristics of mental models are based on observations made by psychologists and cognitive scientists. *However, they can be directly derived from the axiomatic theory of design modeling (Zeng 2002). In the next section, we will describe how to derive a definition of a mental model based on the axiomatic theory of design modeling.*

HCI has a strong relation with mental models. Ideally, the interface of a computer system should be kept consistent with the user's internal mental model about the environment. For example, it is important to design a writer program that has the functionality and appearance that is similar to people's actual writing habits. Normally, there is a plain document like a paper in front of a user. The user inputs words from left to right within this plain document. The user does not need more training to write a simple document with such writer programs.

Norman identifies three models of a computer system as follows (Norman 1988):

1. System model: the actual way that a system works from the programmer' perspective.
2. User's mental model: the way that the user perceives how the system works.
3. Design model: the way the designer represents the program to the user, including presentation, interaction, and object relationships.

However, other researchers propose similar theories but adopt different terms to describe these three models. The next section reviews the theories to help people eliminate such confusion.

Between the design model and the user model is the interface of the system. Of these three models, it is the design model that determines the interface of the system. In other words, the design model guides the interface design (Saja 1985). Furthermore, Norman states that the best way to design an interface is to conceal the system model and indulge the user's mental model. For instance, a driver does not need to understand how a car works but needs only understand how to effectively drive the car. The interface derived from the design model helps the user develop the user's mental model. The resulting mental model will then be a representation of the system, the system's capabilities translated into the user's individual means of understanding. If these two models, the design model and the user model, do not correspond with each other, a problem about how to operate the system arises. After analyzing the logical relations between each part, an important rule for designing a proper user interface can be stated as follows: in order to design a user interface, a suitable design model must be built before the user forms the individual user's mental model. Designers may not be able to build a design model of a nonexistent system by imaging action. On the contrary, designers should create a user's cognitive model, which is the basis of the designer's mental model.

4.1.2 Axiomatic Approach to Analyzing the Mental Model

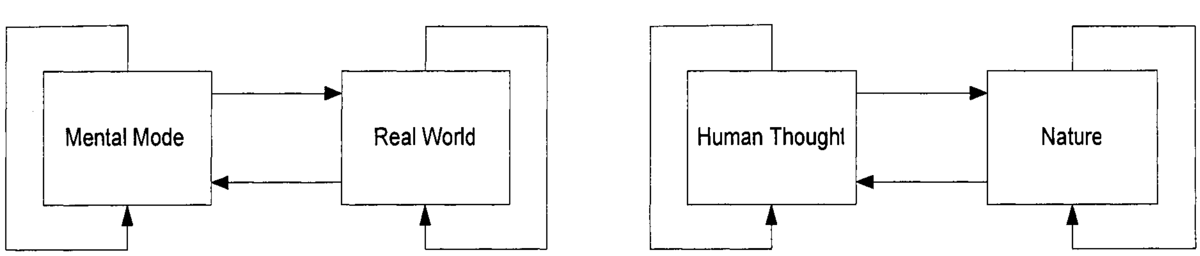
This approach to a human being's mental model is based on the axiomatic theory of design modeling. First, a new definition of the mental model is derived from the axiomatic theory of design modeling. Second, a new viewpoint of mental model's framework is presented. This framework includes many familiar terms that describe different mental models.

4.1.3 Definition of a Mental Model

In this section, a description of a human being's mental model based on the axiomatic theory of design modeling is presented. The five characteristics of mental models (Gentner and Stevens 1983) can be derived from this description.

4.1.3.1 The Axiomatic Theory of Design Model and Mental Model

Based on the axiomatic theory of design modeling, there are four elements to be taken into consideration: the natural system (real world), the human rational system, recognition, and action. Comparing the mental model structure that is presented by A in Figure 6 and the human rational system in the axiomatic theory of design modeling that is presented by B in Figure 6, we find that both of these structures have exactly the same meaning.



A B

Figure 6 Mental Model and Human Rational System

At first this resemblance of the mental model theory and the axiomatic theory of design modeling may be considered a coincidence. However, the following context shows that the axiomatic theory of design modeling can build a formal framework including many important concepts of the mental model in HCI. Furthermore, the axiomatic theory of design model provides a formal cognitive model of a designer that guides UI designers to develop a proper interface for the designer during the conceptual design stage.

With deeper exploration, another interesting result can be found. Not only does the human rational system in the axiomatic theory of design modeling have a structure that is similar to that of the mental model but also all of the five characteristics of the mental model can be derived from the axiomatic theory of design modeling. Table 1 shows that the first, second, and fifth characteristics can be directly derived from the axiomatic theory of design model.

Axioms of the axiomatic theory of design modeling	*Characteristics of mental model*
Axiom 3. Human beings are bounded in rationality	1. They are incomplete and constantly evolving.
Axiom 4. Human beings do not recognize objects accurately	2. They are usually not accurate representations of a phenomenon; they typically contain errors and contradictions.
Axiom 5. Causal relation is the only plausible relation in all relations between causes and effect.	5. They can be represented by sets of condition-action rules.

Table 1 Axioms

The two remaining characteristics can be presented in the following section, which describes how human beings create a mental model corresponding to a specific external system.

In order to recognize an object accurately, three fields should be considered based on the axiomatic theory of design modeling. They are the action input to the object, the action from the object, and the interaction within the object itself. Therefore, the first step is to identify the boundary between the desired object, or the external system, and the human being's thought. It is easy to set a boundary. Human being's thought exists in human being's brain. The biologic constraints of human beings can be the constraints of human being's thoughts. Second, after a clear boundary between a human being's thought and a system is built, an interface between the thought and the system is derived naturally. A human being has to interact with the system by using human abilities, such as seeing, hearing, and touching. Based on Axiom 4, a natural result can be derived: a human being's thought only partially reflects the real world. Moreover, a human being's thought contains mistakes and conflicts, because of the limitation of human beings' abilities. Third, a human being creates perception, selection, and decision by relying on reasoning. This is a fundamental function of a human being's thought. After a human being makes decisions, human beings act upon the system and observe the feedback. They then confirm or adjust their perception of the system. Actually, because of the similarity between the mental model structure and the human rational system structure, the mental model can replace a human being's thought in this paragraph without changing anything. Therefore, based on the understanding of the human rational system in the axiomatic theory of design modeling, a new definition of the mental model can be derived.

In summary, a human being's mental model can be described as an internal symbolic system in an individual's brain. The model is created through interacting with the real world, such as the natural environment, artifacts, and others. These models not only reflect the structure of specific systems but also reflect human being's behavior based on their perception, selection and decision functions.

4.1.3.2 Physical Carrier of Mental Model

Based on the axiomatic theory of design modeling, the mental model structure can be described Equation (17), which follows. Nature and memory can be represented by N and M.

$$\oplus (N \cup M) \tag{17}$$

The physical carrier of the mental model in the human being's mind can be divided into two parts: working memory, M_w, and long term memory, M_L. Each part has different characteristics and functionality. This relation can be described in the Equation 18.

$$M = M_L \cup M_w \tag{18}$$

By combining Equation (17) and Equation (18) and applying the rules of the axiomatic theory of design model, we get the following result.

$$\oplus (N \cup M) = (\oplus N) \cup (\oplus M) \cup (N \otimes M) \cup (M \otimes N) \tag{19}$$

$$= (\oplus N) \cup (\oplus (M_L \cup M_w)) \cup ((M_L \cup M_w) \otimes N) \cup (N \otimes (M_L \cup M_w))$$

$$= (\oplus N) \cup (\oplus M_L) \cup (\oplus M_w) \cup (M_L \otimes M_w) \cup (M_w \otimes M_L) \cup (M_L \otimes N) \cup$$

$$(N \otimes M_L) \cup (M_w \otimes N) \cup (N \otimes M_w)$$

A specific system, N, can be decomposed into two parts, O_i and O_r. O_i represents such objects that can be observed. O_r represents such objects or relations within a system. Hence, $N = O_i \cup O_r$

Psychological experiments show that people complete all kinds of cognitive processes only in working memory whereas long term memory, acting like knowledge storage, supports human beings in memorizing things for a long time (Newell and Simon 1972). Based on the characteristics of memory and the process by which a mental model is created, the mental model can be classified into four levels.

1. The first level is a simple image. People create an image corresponding to the real things' physical geometry. These images are stored in the visual memory contained in the working memory at first, and they are then transferred to long term memory.
2. The second level is a mixed level. At this level of the mental model there are ambiguous relations between each object. Actually, this mental model is a transition between level one and level three.
3. At the third level there is symbolic notation. People develop symbolic notation to represent the real thing's structures or relationships. Symbolic notation can be a language

or image-symbol notation. Individuals have a distinct mental model depending on their knowledge and abilities. If people understand only the functions of the system, a basic mental model that contains only the causal relations between objects is enough. While this level of the mental model is completed, we can state that people already abstract this external system. An abstract mental model is a simplified explanation. The third characteristic is reflected at this level of the model.

4. The fourth level is a cognitive level. People engage in perception, selection and decision based on theories of logic. This level can be described as the cognitive model.

4.1.4 Framework of Mental Model

In order to explore how a mental model affects the development of an interface between human beings and a system, researchers in the HCI domain have created several terms. This section focuses on how to describe these conceptions based on the axiomatic theory of design modeling.

4.1.4.1 Definitions

Before we apply the axiomatic theory of design modeling to describe the framework of a mental model, a list of symbols and the corresponding meanings need to be presented.

U — user, person who operate the system.

S — system, the user's operand..

E —environment, the environment in which the user operates the system.

D —designer, person who design the system for the specific user.

The four elements above create three systems: the user system, the product system, and the mental model system.

User system

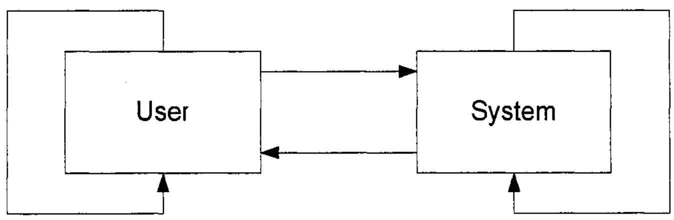


Figure 7 User System

There are two elements in the user system: User and System. The structure of the user system can be given in the following equation:

$$\oplus U = \oplus(U \cup S) \qquad (20)$$

$$= (\oplus U) \cup (\oplus S) \cup (U \otimes S) \cup (S \otimes U)$$

Actually the equation above has so many meanings that we are not able to explain all of them one by one. In fact, it is not necessary to do that. In this thesis, we pay attention only to a human being's thought. So a user model can be described by such an equation.

User model:

$$(U \otimes S) \cup (S \otimes U) \qquad (21)$$

Each user holds an individual representation of a specific future system.

User preliminary mental model:

$$\oplus U \tag{22}$$

Each user holds an individual representation of a specific existing system.

System model:

$$\oplus S \tag{23}$$

An accurate and comprehensive model describes the system.

Product system

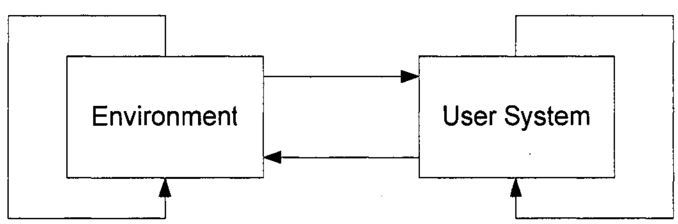


Figure 8 Product System

The product system expands the user system by including an environment element. The structure of a product system is explained in Zeng's paper (Zeng, Pardasani et al. 2004).The following equation describes the structure of a product system.

$$\oplus P = \oplus(E \cup Us) \tag{24}$$

$$= (\oplus E) \cup (\oplus U) \cup (E \otimes U) \cup (U \otimes E)$$

Mental model system

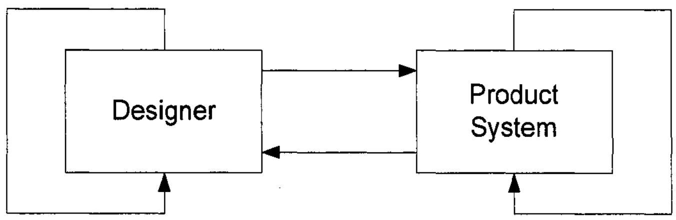


Figure 9 Mental Model

This system, which is built on the top of user and product systems, describes designers, the product system, and their relationship. The following equation describes the structure of this system.

$$\oplus M = \oplus(D \cup P) \tag{25}$$

$$= (\oplus D) \cup (\oplus P) \cup (D\otimes P) \cup (P\otimes D)$$

Based on this equation, two important concepts can be derived.

Designer model:

$$(D\otimes P) \cup (P\otimes D) \tag{26}$$

The designer holds a general model of a specific system in his/her mind.

Conceptual model (Designer model $\cup$ User model): There is a reference to the conceptual model in Norman (1983) about conceptual model: "Conceptual models are devised as tools for the understanding or teaching of physical systems. Mental models are what people really have in their heads and what guides their use of things". In other words, the designer designs a conceptual model into the system so that it appears graspable and coherent to the user. If the users manage to get the conceptual model right, the correct user mental model will follow.

Cognitive model:

$$f^c((\oplus(P \cup U)) \quad (27)$$

Cognitive models are models built in the people's mind to reflect the product system at the logical level.

Figure 10 shows the framework of mental model based on the axiomatic theory of design modeling, proposed in this chapter. It explains two different cases. The first one is that if the system has already existed, users can create their individual user models by communicating with the system model. In this case, their cognitive models can do nothing to change the system model. The second one is that if the system is a future product, the system designers create their design model mainly based on the user's cognitive model. In the context of sketching user interface design, the user's cognitive model is the mechanical designer's cognitive model. This cognitive model plays a critical role during the design process of the user interface.

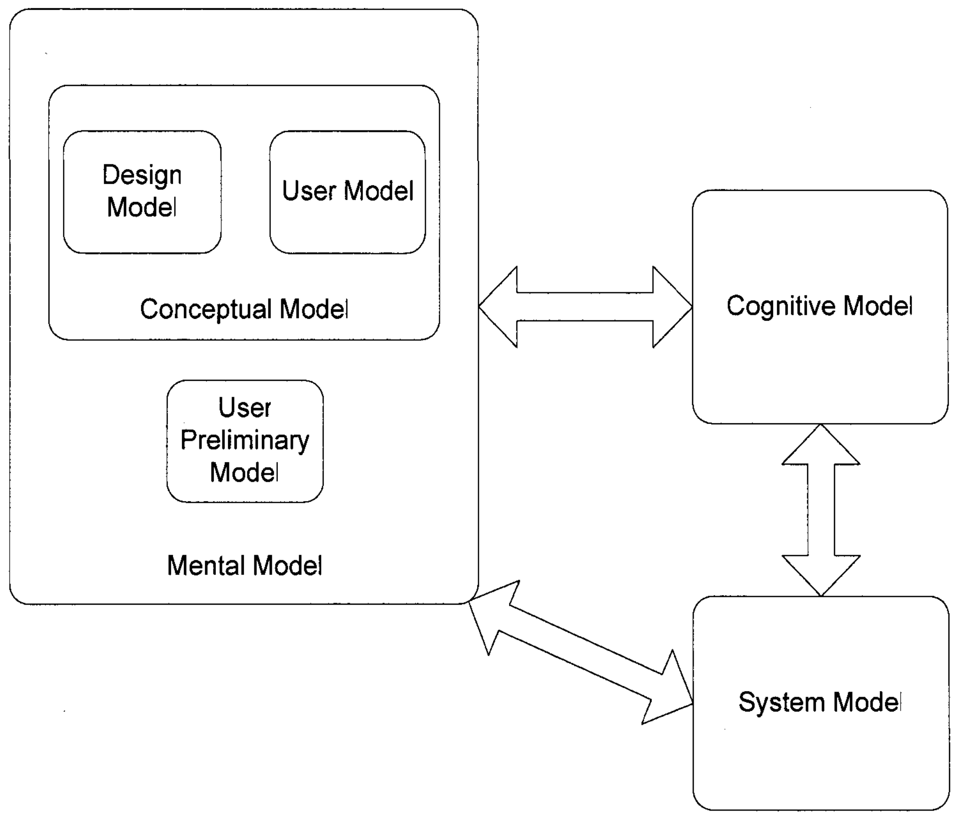


Figure 10 Framework of Mental Model

The next section of the present thesis relates the work done by other researchers in this field in terms of the framework of the mental model given in Figure 10.

4.1.4.2 Relation to Existing Work

Since the Scottish psychologist, Craik, firstly introduced mental models in 1943 (Craik 1943), researchers from psychology and cognitive science have continued his effort and proposed a set of terms to describe more detailed states of a human being's mental model. Unfortunately, a uniform standard about how to use them does not exist. Hence, people often find that the same terms in different books have different meanings or they find that different words mean the same thing.

To eliminate the reader's confusion about these terms, we review the history of mental models.

Craik stated that a mental model, which is constructed in a human being's mind, is used to anticipate events, to reason, and to underlie explanation (Craik 1943). The term is commonly used to describe how human beings model the real world in their minds. Following Craik's work, many researchers have developed their own definitions of mental models. For example, Carrol and Olson state that a mental model is : "a representation (in the mind) of a physical system or software being run on a computer, with some plausible cascade of causal associations connecting the input to the output" (structuring user interfaces with a meta-model of mental models) (Carroll and Olson 1988). Staggers and Norcio define mental models as organized structures, consisting of objects and their combinations (Staggers and Norcio 1993). Redish concludes that mental models consist of propositions, pictures, procedural rules and statements and indications of how and when they may be used (Redish 1994). Rauterberg proposes that mental models should contain three different kinds of knowledge: the structure of the logic of the work, the sequential structure of the targets and the temporal structure of all operations (Rauterberg 1995). Sifaqui suggests that a mental model is the manner in which the user carries out his or her activities (with or without a computer) and how he or she organizes the information in his or her memory (Sifaqui 1999).

It can be seen from the review above that confusions exist in the definition of mental model due to the lack of uniform terminology. Norman and Yong were attempt to organize these terminologies (Martina 1997).

Norman introduces the distinction between four different types of mental models that exist in the area of user-system interaction. The structure of mental models is illustrated in Table 1.

<table>
<tr><th>Norman</th><th>User</th><th>Design</th><th>System</th><th>Scientist</th></tr>
<tr><td>1983</td><td>Mental Model</td><td>Conceptual Model</td><td>Target System</td><td>Scientist's Conceptualistation</td></tr>
<tr><td rowspan="2">1986</td><td>User's Model</td><td>Design Model</td><td rowspan="2">System Image</td><td rowspan="2"></td></tr>
<tr><td colspan="2">Conceptual Model</td></tr>
</table>

Table 2 Norman's Mental Model

The computer system with which the user is interacting is the target system. The user constructs a user's mental model of that target system. The target system may communicate with the conceptual model, which is a representation of the target system within the designer's mind. Users build the user's mental model through interacting with the target system. The scientist's conceptualization is a model describing the content and structure of the user's mental model. In 1986, Norman revised his theory of mental model. In his revised theory, the designer builds a design model that is communicated through the system image. The user develops a user's model through interaction with that system image. The design model and the user's model together form the conceptual model of the system.

In the knowledge of the internal structure of a system, DiSessa proposed two kinds of models of systems (di Sessa 1986): *Structural models* provides a complete and predictive model based on a system model. This model is a detailed understanding of the system. In contrast, *functional model* is an abstract model of a system model. The functional model represents selected properties of the system in terms of a specific task the system is called upon to perform.

Nielsen attempted to introduce mental models into HCI and proposed a meta-model to classify models of the user-system interaction (Nielsen 1990). There are seven elements or models in this framework.

U — users;

D — designers;

C — the computer system;

M — manuals and other documentation of C;

T — the tasks performed by the user;

W — the surrounding world in which U performs T;

R — the researcher looking at any of the above;

A simple notation can combine these elements to describe different representations. For instance, a user's model of a computer system is UC; a design model of a computer system is DC.

Comparing our framework and the one above proposed by Nielson, we find that both structures are similar. *However, the obvious difference between these two frameworks is that there is another user model in this framework: the user's preliminary model.* This term makes a distinction between the future system and the existing system. This system describes the human beings who used to process the tasks. Before the future system is designed and developed, people have hoped that the future system will be an improvement. Therefore, there are two user's mental models corresponding to each system. Because user's mental model is widely adopted by most researchers to describe a mental model of a future system held by users, the user's preliminary mental model is introduced in this thesis to describe the mental model of an existing system held by users.

4.2 User's Cognitive Model

Because this thesis on developing computer-aided software for mechanical designers, two kinds of designers must be taken into consideration: a designer who designs the software and a mechanical designer who actually is a user of this software. It is important to keep these two designers distinct. Therefore, the designer's cognitive model refers to the software designer's cognitive model while the user's cognitive model refers to the mechanical designer's cognitive model. Before developing a proper interface for mechanical designers, it is necessary to explore the user's cognitive model that is the mechanical designer's cognitive model.

There are many definitions about design. From the mental model viewpoint, when a designer tries to design an inexistent product to satisfy the user's requirements, a mental model of this product should be created in the designer's mind. In one word, a designer's cognitive model is a reflection of a future product system as Equation (27) (i.e., $f^c((\oplus (P \cup U))$ describes. As Section 4.1.3.2 indicated, there are four levels in the mental models of a user model. The most important task of the designer's cognitive model is to describe how to build these four levels of user mental models. The following describes such a process. While exploring the cognitive model of designers, the functions of sketching are also uncovered.

Designers must create a mental model of the future product based on the requirements. The designers' cognitive model has some specific characteristics compared with the general cognitive model of human being.

1. Generally, because the future product does not exist, designers cannot create the first level mental model from any external system. That means designers have to depend on external tools to build the mental model. These tools can be sketches or diagrams.

2. Because designers need to create a mental model of the new product, designers would have more creative activities than in the general cognitive model.
3. Unlike the general cognitive model, designers have to rely on different patterns of logic to achieve the design objectives. People depend on general cognitive models to understand the world and then use perception based on the three patterns of logic: deduction, induction, and abduction. However, based on the axiomatic design theory, a design problem is an ill-defined problem. The solutions to the problem and the definitions of the problem evolve simultaneously in the design process. Designers have to rely on a different pattern of logic: recursive logic to solve design problems. That is an important difference between the general cognitive model and the designer's cognitive model.
4. When designers build a mental model in working memory, the external tools also need working memory. Because working memory is a limited resource, designers need to reduce the amount of working memory that is to external tools.

The most important responsibility of designers is to create a mental model of the new product. Designers need to create all four levels of the mental models.

At first, designers have only a rough description of the new product. The description includes different requirements imposed on the new product. Designers analyze the requirements. These requirements can have various formats, such as the use of natural language, and of sketches. Designers read these requirements and start to identify which is product and which is environment to create a conceptual model. This conceptual model can give a rough picture of the new product. Designers need to set a rough boundary between product and environment. Then designers can decompose the environment into three environments: natural, built, and human environment. Designers need to analyze the relationships between sub-environments and product as well as between two sub-environments. For each relationship, sub-requirements can be derived.

With more and more sub-requirements emerging along with the decomposition, designers manage to create a rough model of the new product. At the same time, because most designs are redesign, there is already an existing product. What designers need to do is to modify this existing product to meet the new requirements. Using their insight concerning the new product, designers classify the new product into a series of existing mental models that can be called experience or knowledge. Or, designers combine different mental models to make up an image of the new conceptual model of this new product. These conceptual models are mostly general concepts, such as software and machines.

Because a mental model contains the four levels of abstraction, designers try to describe each part of the new product that interacts with different environments. A rough sketch describing the geometry of the product and sub-objects within this product is created by designers. This sketch is just like the first level mental model. However, even this sketch contains only very simple information, since designers actually have no idea about any solution. This sketch needs to be adjusted for deeper exploration. At the end of this process, designers obtain even more specific requirements imposed on the new product.

The whole mental model of a product may contain models of four levels. Designers have only a rough image that contains several sub-objects derived from the above analyses. These specific requirements can be considered as q in the pattern of recursive logic. If designers have $p \rightarrow q$ which can be considered as rules, they can say that the exact p can be the right solution. Unfortunately, designers cannot know the exact rules at first; therefore, they also have no ideas about solutions. How do designers resolve this challenge? Based on the axiomatic theory, designers should evaluate the problems and problem solutions at the same time. To meet all these requirements, designers can classify them into two categories. The first one is that designers already know there is a rule or several rules that can be followed. Following these rules designers can create satisfying solutions that meet the requirements. In the other words, designers can rely on their experience or on the existing mental models to find solutions for these requirements. However, designers cannot seek any existing mental models that can meet the second kind of requirements. Actually, to solve the second kind of requirements is the core responsibility of designers. There are two ways to solve these problems. Generally, although designers have no satisfactory solutions for the requirements, they may have some similar solutions. These solutions can become the first stage of the eventual solutions. Designers can be guided by some principles, such as TRIZ, to modify these solutions and then rely on the rules of logic to get a logical result. After evaluating these logical results, designers come back to readjust these solutions. The whole process continues until the desired result is obtained. When a desired solution is found, a part of the product may move from the new product and become a part of the environment. Then designers analyze the rest of the product and the updated environment again. This process is repeated until what remains is reduced to zero. Another situation is that designers rely on inspiration to jump from one solution to another solution without the interim stages. To achieve the desired result, designers redefine the total level models. The following figure describes the design process.

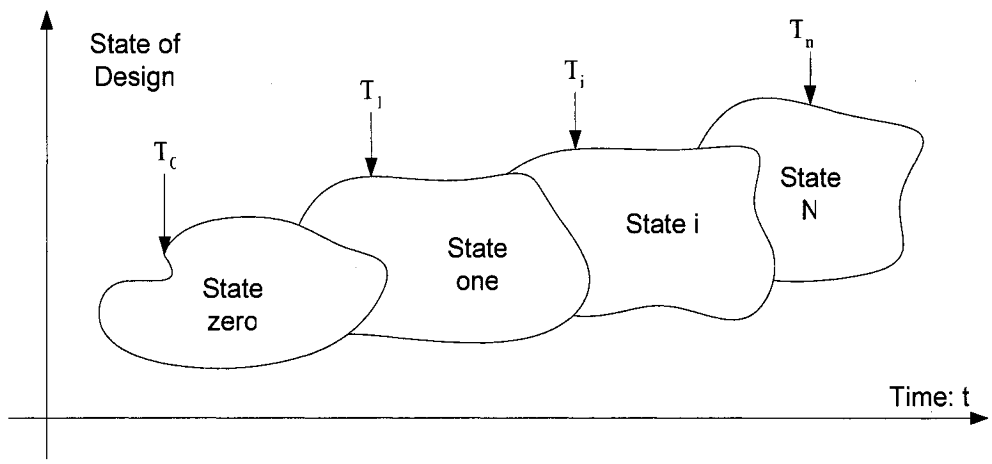


Figure 11 Design Process (Zeng 2002)

Normally, most designs are redesign. Designers seek existing mental models and try to catalog the conceptual model into these catalogs. This process can support designers to continue their remaining work.

Generally speaking, designers analyze the requirements at first. These requirements can be presented in various formats. Language is a common medium. Designers read these requirements and seek existing mental models to get a similar mental model. Through these requirements, designers can build rough mental models of the future product. The rough mental models may include level one and level two. An important responsibility of designers is to create advanced mental models: third level and fourth level. Distinct design theories guide designers to implement this process.

4.3 Requirements on Sketching System for Conceptual Design

Based on the characteristics of sketching process introduced in Chapter 2 and the framework of mental model proposed in this chapter, the following main results can be found about the designer's cognitive model and the necessary requirements of a proper sketching system for designers:

- First, sketches evolve progressively in the design process and the design process is a successive process avoiding any outside intervention. The designer's cognitive model is a process that develops a user mental model in the working memory, however the working memory is a limited resource, and any additional requirement of working memory reduces the effectiveness and efficiency of the design.
- Second, attempting to recognize a sketch within a design process offends the important characteristic of conceptual design, which is that design solutions and design problems develop simultaneously in the conceptual design stage. This characteristic of the design process indicates that the preliminary design solution is ambiguous. Therefore, a sketch which presents a solution for the design problem can not be recognized accurately by software or by other designers. Any attempt to recognize the sketches requires the designer to adjust the recognition results. This

requirement conflicts with the first result, which is that the design process can not be interfered with.

- Third, because sketches evolve progressively in the whole design process, a computer-aided sketching system must not only provide an environment for sketching but must also be a powerful tool to aid designers to organize the sketches with the corresponding design requirements.

Chapter 5
Software Prototype of Interface

5.1 System Requirements

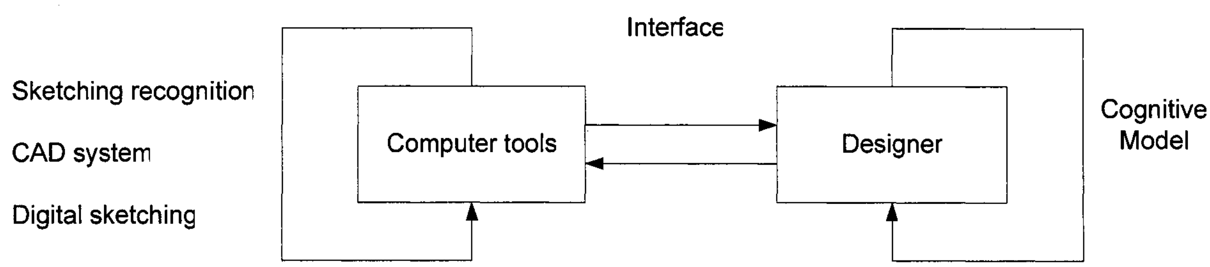


Figure 12 Sketching System Architecture

The preceding chapter suggests that there are three necessary requirements to develop a good sketching system for designers in the conceptual design stage. Based on the results above, we can divide up a sketching system into three parts as shown in Figure 11. The first requirement is to summarize the characteristics of the cognitive model of the designers. The second requirement is to design a proper interface between the designers and the sketching system. The last requirement is to choose proper computer technologies that provide the fundamental functions to the interface, such as sketching recognition.

Chapter 4 discusses the first requirement of a sketching system. A series of characteristics of a designer's cognitive model are summarized at the end of Chapter 4. Based on the characteristics of a cognitive model, Chapter 5 proposes a natural interface that satisfies the second requirement. In order to achieve our final goal, developing a natural interface, some computer technologies are introduced. However this thesis does not focus on how to complete these technologies.

5.2 Perceptive User Interface

The analyses given in Chapter 4 indicate that there are two obstacles that prevent users from successfully operating a specific system through an interface. The first one is that the design mental model and the user mental model may not match very well. In order to overcome this obstacle, a cognitive model of a user, which describes the process of developing a user mental model, is provided to aid designers to understand their future users. With the help of the cognitive model of a user who is a mechanical designer, UI designers can not only develop an appropriate interface but can also evaluate the interface. Another obstacle is that the user's preliminary mental model is quite different from the user's original mental model. A specific term called consistency is used to describe this situation (Saja 1985). Consistency is particularly important in the HCI domain. There are two levels of consistency in HCI. The first one exists between the processing of two similar tasks. For instance, people always tend to press "ctrl" and "s" keys to save documents while working on a Windows platform. Another one exists between the primary user mental model and the later user mental model. This consistency is mentioned by only a few researchers. However, with the development of computer technology, more and more researchers will take part in this field.

To their interactions with technology, people bring attitudes and behaviors similar to those that they exhibit in their interactions with each other (Turk and Robertson 2000). In other words, people always try to apply their familiar mental models to similar situations without any additional effort. However, current computer interfaces, used for office applications, are primarily functional rather than social. Computers have their own specific ways of behaving. They change our usual behavior. There are significant differences between two mental models. Since the first computer came into existence, many researchers in HCI have endeavored to eliminate this difference. They simply want to help users operate computer systems more easily.

Table 3 shows the progression of major paradigms in HCI. When people first adopted computers to handle tasks, there was no big difference between users and designers. Most systems were designed by programmers and also used by programmers. The importance of the interface was not very significant. Later, command line interfaces became popular and nobody wanted to use punch cards any more. About ten years ago, WIMP-based graphical interfaces (using windows, icons, menus, and pointing devices) were introduced into people's lives. When people look through the whole history of the development of interfaces, an important rule is revealed, that all of people's effort is to release user's mental load. One interface replaces another one, because users think the later interface is more efficient and be easier to use.

Era	*Paradigm*	*Implementation*
1950s	None	Swithces, wires, punched cards
1970s	Typewriter	Command-line interface
1980s	Destop	GUI/ WIMP
2000s	Natural interaction	PUI(multimodal input and output)

Table 3 Evolution of User Interfaces (Turk and Robertson 2000)

In recent years, researchers have been discussing post-WIMP interfaces and interaction techniques. It is expected to require integration at multiple levels of technologies such as speech and sound recognition and generation, computer vision, graphical animation and visualization, language understanding, touch-based sensing and feedback, learning, user modeling, and dialogue management. This next major paradigm of HCI that is more natural and powerful is called the perceptual user interface, which takes advantage of both human and machine perceptual capabilities. Based on this paradigm, the gap between the primary user mental model and user mental model can be minimized. An ideal result would be that users would not need more training with the new interface and that they could switch to a new computer system without any difficulty.

There are three types of PUI. The first one is a perceptive UI adds human-like perceptual capabilities to the computer. For example, it allows computers to interact with what the user is saying or what the user's hands or body are doing. Such interfaces provide another type of input that is absent in the current interface design. The second type of PUI is a multimodal UI. The multimodal UI emphasizes human communication skills. Most work on multimodal UI has focused on computer input, such as using speech, pen-based gestures. The third type of PUI is a multimedia UI that uses perceptual and cognitive skills to interpret information presented to the user. Text, graphics, video and audio are the typical media used.

Researchers in the field of interface development have already brought out several new systems. The QuickSet system developed by Oviatt and Cohen is a computer system integrating speech and pen gestures as users create and control military simulations. This integrated system shows how multiple modalities can create a more natural and stable system.

Perceptual user interface models traditional human interaction between humans and their physical environment. Unlike the current popular user interface, PUI may enable users to interact with computers in a more natural, efficient, and easier way. Users, despite their previous training, can switch without difficulty to a computer system based on PUI. Based on the analysis of a mechanical designer's cognitive model and PUI, an important rule that one can apply to the development of an interface is to use a PUI that is the most appropriate interface for designers.

5.3 System Architecture

Our system inherits from PUI and satisfies the above design requirements in our attempt to develop a sketching system for designers involved in the conceptual design process. This system contains two parts: software and hardware. The functions of current hardware are so powerful that we do not need to make any improvements. So we focus on how to develop the software to aid designers. Our standard hardware contains a Tablet PC and a light pen as shown in Figure 12. Designers can also use a mouse instead of a light pen when it is necessary.

Figure 13 Tablet PC and Light Pen

Briefly, the software can be divided into three types based on MVC design pattern. The following figure describes the structure of this application. The sketching control model works as the Controller part. It provides the basic environment of the sketching system. Its responsibility is to manage the other two parts and fulfill the proper actions depending on the received commands. In this model there are three sub-parts: operation management, gesture command management, and sketch management in this model. There are two kinds of commands in the system. One is the traditional command, using menus and buttons. The other kind is the gesture command. Designers can easily input a gesture command by drawing a gesture by pressing the right button of a mouse or a light pen. If the system switches from the traditional command state to the gesture command state, the ANN recognition engine is invoked to verify the input gesture commands. The sketch data model works as Data part. This model is made up of two sub-parts. The first part uses a sketch data file to store the information about a sketch. The other part uses XML gesture data file to store the information from an inputting gesture command. The following figure presents the whole structure of this system.

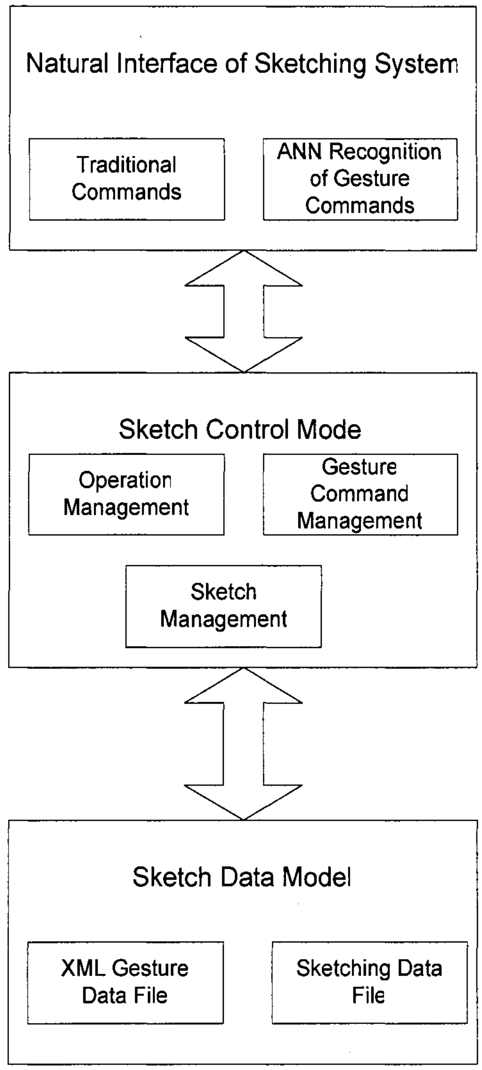


Figure 14 Architecture of Software

There are two important parts in our software. One is the data model that deals with sketch data structure. The following section discusses two sub-models involved in the data model: operation management and sketch management. The other part consists of a gesture command recognition model that, by using a neural network, deals with how to recognize a designer's gesture commands.

5.3.1 Gesture Command Recognition Model

Gesture command recognition has a strong relationship with sketch recognition. However, sketch recognition generally uses more complicated algorithms than gesture command recognition. The following section gives a brief introduction to sketch recognition that provides a basis for understanding gesture command recognition, although this sketching system does not include any sketch recognition algorithms.

5.3.1.1 Introduction of Sketching Recognition

There are two different sketches depending on the ways of inputting sketches. One is on-line sketching; another is off-line sketching. The difference between these two methods is that on-line sketching is an interactive process and off-line sketching is that sketches are input as images. Designers use a pen or a pen-like device to draw sketches on the screen directly. The sketching system provides dynamic interpretations during the interactive process. We focus on the on-line sketching because it gives users immediate feedback.

There are three different levels of sketch-based UI depending on the inputting patterns and stroke orders. The first level of the application requires that the stroke sequence be fixed as in an application for handwriting recognition. The second level application is like domain specific diagram constructing tools that ask users to have a common knowledge background. The third level is a more general and more difficult one (Landay and Myers 2001). Different designers have their own input styles to express their thoughts. A free-hand sketched-based system is such a case.

Extensive research on on-line handwriting recognition has been done. It is common for users to use many maturate applications for handwriting recognition in the real world. However the applications for on-line sketching recognition are usually in the lab. Researchers in different fields have endeavored to invent several approaches to improving the accuracy of on-line sketching recognition applications.

The process of recognition of on-line sketching can be briefly divided into two procedures: preprocessing and shape recognition. A preprocessing process contains two steps. First, the stroke is polygonalized to an open polygon or polyline. Second, the noises of the stroke may be polished. To deal with shape recognition, many researchers propose a series of approaches. Now the research into on-line sketching recognition can be classified as following: (Liu, Cha et al. 2002)

a) The first method is based on linear least squares fitting to a conic section equation.
b) The second method employs fuzzy logic to identify a sketch via the sketching position, direction, speed and acceleration (Qin, Wright et al. 2000; Qin, Wright et al. 2001; Qin, Wright et al. 2001)
c) The third method is based on neural network. Different input values extracted from geometry features are fed forward in neural network. Wei introduced gravity radii and regularized gravity radii (RGR) and used RGR as parameters of BP neural

network(Liu, Cha et al. 2002). Suzuki proposed a fuzzy pline curve identifier (FSCI) and the fuzzy neural network is trained (Suzuki, Itakura et al. 2001). Figen proposed a recognizing method for extracting the internal angle feature of shape and using a binary synaptic weights algorithm (Figen, Andrew et al. 1996).

Because the present thesis applies the neural network method to recognize gesture commands, the following sections present an algorithm to recognize gesture commands based on neural network. Actually, many similar methods have been proposed in recent years, but the emphasis of this thesis is not on comparing these complex neural network algorithms. Therefore, a simple algorithm that can be easily implemented has been adopted in this thesis.

5.3.1.2 Introduction of ANN

Neural Network is a field crossing Artificial Intelligence and Approximation Algorithms. The NN is modeled on a human being's brain. It has a multi layered structure. Each layer consists of several neurons.

A neuron has the following structure:

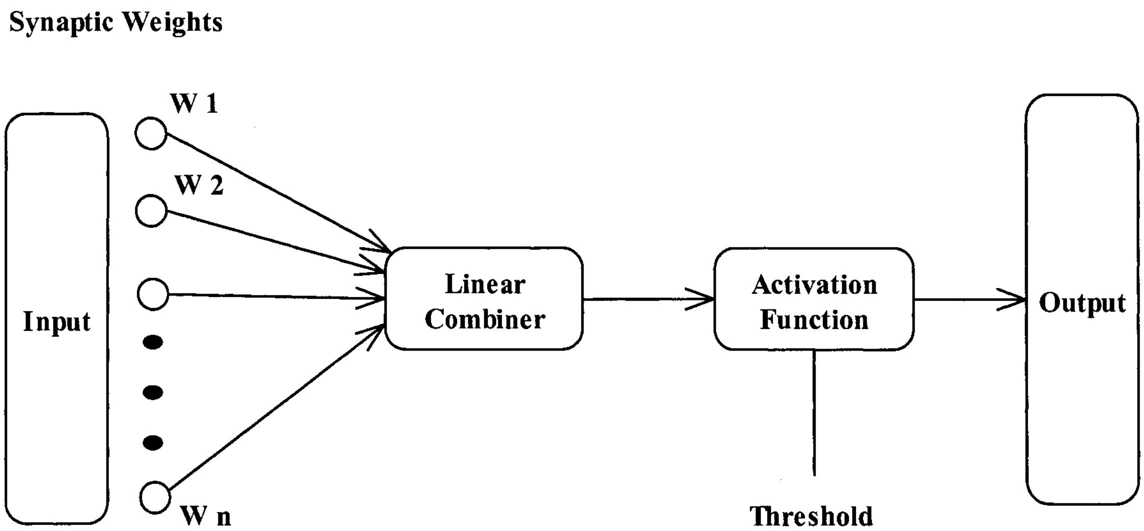


Figure 15 ANN Structure

The neural network consists of multiple layers that are composed of numbers of neurons. The following example represents a neural network with three layers:

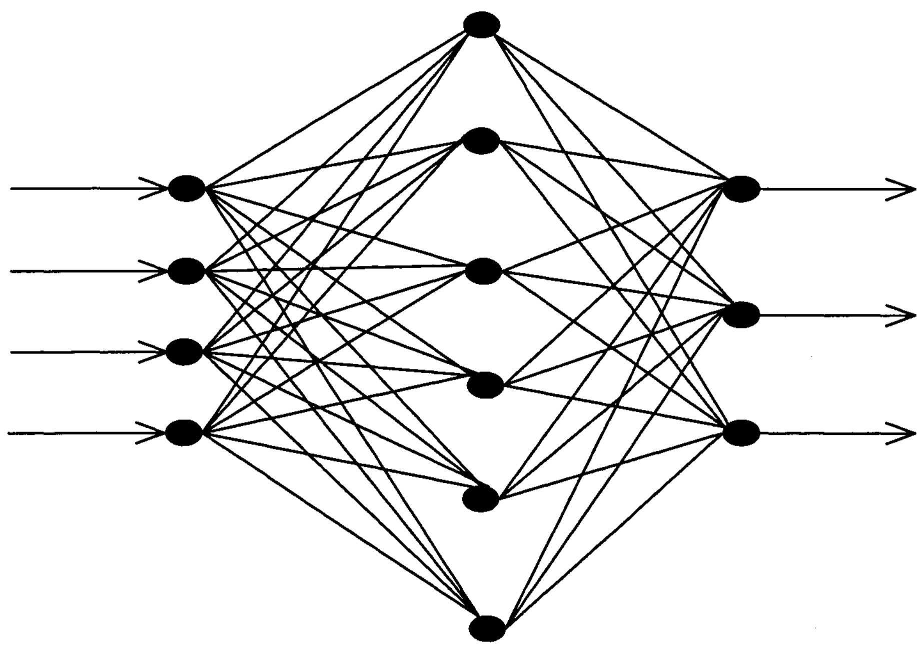


Figure 16 Layers of ANN

This NN receives 4 inputs and produces three outputs. In the case of an image recognition application, the inputs would be pixels. The outputs would be produced by an activation function, after the first layer of the neuron receives the inputs. Then the outputs become the inputs of the next layer until the last layer is reached. We can adjust the neurons to work better with particular inputs by adjusting the learning rate. Once we decided what adjustment we need to do to the neurons in the output layer, the changes to the previous layer will be back propagated (There are many algorithms. One of them is BP which has been used very popularly). This process can be called training. In summary, after the input pattern is presented to the network and processed layer by layer, the difference between what we want and what we get would be used to adjust the network. If the outputs of the trained network are acceptable, the parameters of the network are saved to be used for real data.

5.3.1.3 Introduction of Mouse Gesture Recognizer on ANN Approach

This program is written in C#. It uses a neural network to analyze mouse movement patterns. Because an algorithm with enough tolerance and the ability to learn is required, a feed forward neural network with a back propagation learning algorithm is adopted.

When users draw gestures on the screen, lists of points are recorded. Then these points are normalized to a fixed number of anchor points. Sinus and consinus of a line that is formed by two sequent points are used as input values of the neural network.

There are three layers in the neural network. The input layer of this network includes fourteen neurons. The same number of cells is included in the hidden layer. As in the case of most neural networks, only one hidden layer is included in this network. There are two ways to deal with the output layer. One way is to identify each gesture command with only one output neuron in the neural network. Another way is to identify each gesture command with a corresponding neural network as well as one output neuron in each neural network. The second way is more complex, but the performance is better than in the first. So we use the last one to recognize gesture commands.

5.3.1.4 Algorithm of Gesture Recognition

Gesture recognition is divided into four stages, which are Primitive Gesture, Regularization, Training Neural Network and Recognition.

Primitive gesture

The system predefines a set of gesture commands, such as delete, redo, undo, group objects and ungroup objects gesture commands.

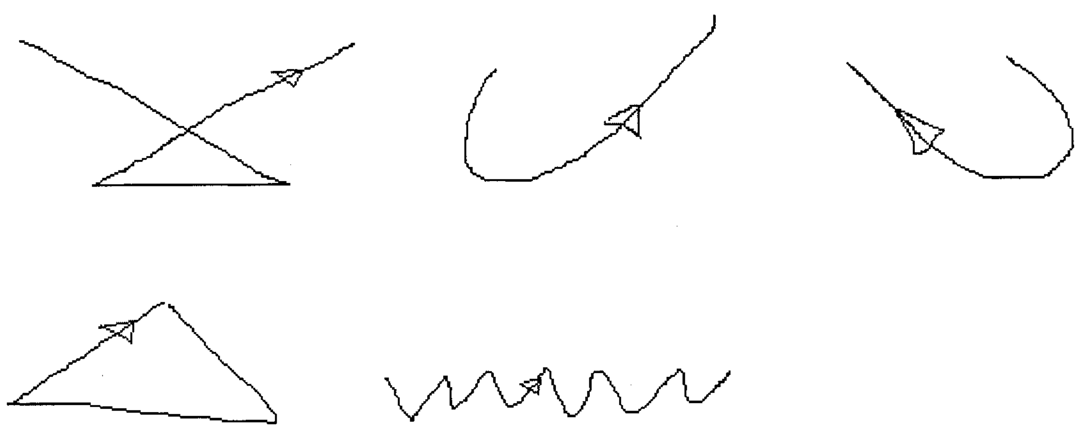


Figure 17 Gesture Commands

Designers can also create and add their desired gesture commands into this system.

The system identifies the primitive gestures according to their features. Designers draw sketches through the interface of the Sketching System. The system captures a set of sequential coordinates. The neural network cannot directly identify different gestures by these coordinates. It is necessary to extract the primitive gesture's feature into a standard form. There are different ways to extract the gesture's feature. One is RGR(gravity radii) (Weinstein and Graves 2002). In this Sketching System, a simple technique is used to extract the feature. At first, fifteen points are selected from the stroke of the gesture then the sine and cosine of the line between two neighboring points are extracted as the feature of the gesture.

Regularization

Because one gesture that needs to be identified commonly includes different numbers of coordinates compared with the primitive gesture, the number of coordinates of the gesture that is input into the system must be regularized. The following steps are adopted in order to make the same number of coordinates (Weinstein and Graves 2002).

1. Calculating the distance between each neighboring pair points.
2. If the number (m) of coordinates of the gesture that needs to be identified is larger than the number (n) of coordinates of the primitive gesture, the nearest m-n points are deleted.
3. If m < n, do n-m times to find out the farthest two neighboring points, then insert n – m points.

Training Neural Network

The training samples can be created automatically or manually. If designers create training samples automatically, the system adjusts the points of the primitive gestures randomly within a limited scale. If designers create samples manually, these samples must correspond to the primitive gesture one by one. After the system finishes training the neural network, all the parameters will be saved to a file.

Recognition

The system loads the parameters of a trained neural network before it identifies a gesture. The neural network is constructed with the loaded weights and biases. All the outputs of these neural networks are fed in a classifier to identify which gesture is the best-matched gesture.

5.3.2 Data Model

The requirement of data structure of the sketching system is efficiency and flexibility. The basic requirement is to store and describe the sketch. This data structure should store all of the necessary information of the sketch. The second fundamental requirement is to fulfill all the basic functions based on the data structure. The third requirement is to achieve the future functions and not to modify the data structure. Operation management and sketch management are involved in this data model. Operation management takes charge of organizing all operations such as ungrouping, grouping, and undo. Sketch management takes charge of organizing different parts of a sketch. Both of them work harmoniously to provide the solid basis of this system.

The sketching system is a vector graphics application and is written by C# language under .Net framework. This application is built under Microsoft's GDI+ library. Because of its advantages of flexibility and easy maintenance, ArrayList, a system data structure in C#, is adopted as the basis of the application. As a vector graphics application, the Sketch application needs to record each point that the user draws on the canvas. Generally the canvas of our application is a windows form.

There are two data structures in the system. The one in operation management is to describe the operations of designers. The second in sketch management is to present the data of sketches. It is described as follows: each stroke is composed of a series of continuous points. Points are the basic data and the smallest elements of our application. Each point has two coordinates: x and y. When users press the left button of the mouse, move the cursor on the screen, and then release the button, a new stroke is created. The attributes of a stroke depend on the pen width and the pen color. The critical element in the sketching system is the object. An object contains not only a series of strokes but also other objects and additional information to that describes how to organize these sub-objects. The first step is to generate the primitive objects by aggregating the related strokes. Then designers can create design sketches to any abstraction level by aggregating the related objects. Because each object contains information about how to organize sub-objects, the reverse process, decomposing the objects into any level of details can easily be achieved. This data structure can powerfully support merging and breaking operations.

Point:

Points are the basic data and the smallest elements of our application. Each point has two coordinates: x and y.

Stroke:

One stroke is composed of a series of continuous points. When users press the left button of mouse, move the cursor on the screen then release the button, a new stroke is created. The attributes of a stroke are pen width and pen color.

Object:

The object is the basic element during the action. An object is composed of one or several strokes or other objects. All the strokes are stored in a list. Another list stores the history of the sub-objects. This information is used when a breaking operation is invoked.

Object List:

All the objects are stored in a list. Some actions, such as draw, cut, paste and delete, can modify this object list. All information in an object list can be stored in a sketch file.

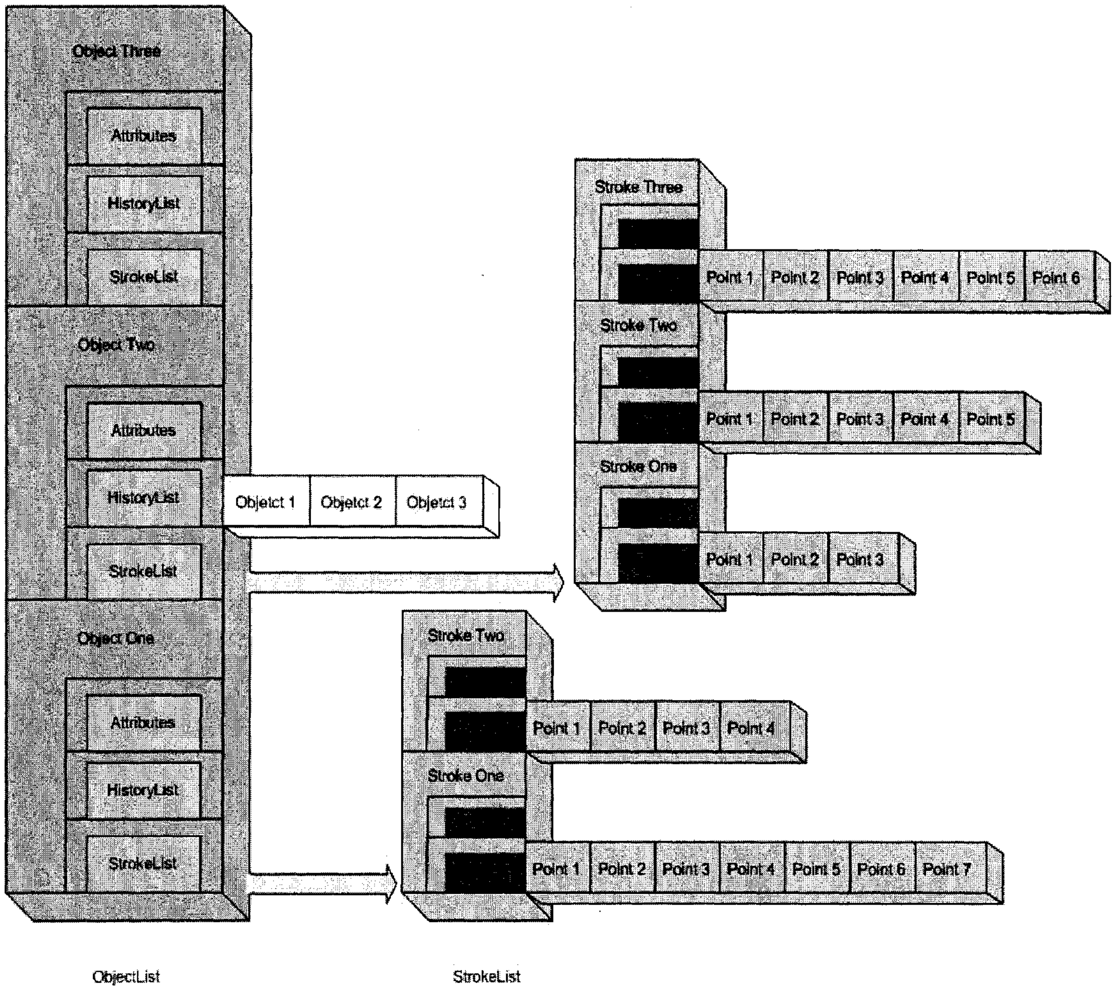


Figure 18 Object Structure

Another structure describes how to organize a designer's operations. It is explained as follows.

Action:

An action is an abstract concept in our application. An action represents an operation that users apply through the interface. One general action could be sub-divided into: select action, unselect action, copy action, paste action, cut action, delete action, break action, split action and merge action. Each sub-action includes redo and undo functions that can be overridden.

Redo and Undo Action Stacks:

This is a data structure using two stacks to store the history of the actions. This structure can perfectly fulfill redo and undo functions. Not all the actions are pushed into stacks. Only Paste, Cut, Delete, Break and Merge actions that can modify the object list are stored in stacks. One stack is to store actions to fulfill the undo function and another is to store actions to fulfill the redo function. Once a new action occurs, this action is pushed into the undo stack. If users undo the top action of the stack, this action is moved into redo stack.

5.4 Sketching Case

This example shows how to formalize a design sketch with our system. The first step is to allow designers to draw freely any sketch in the drawing area. They can use copy, paste, redo, and undo operations to achieve their goal.

5.4.1 Gesture Commands

This case shows how to use defined gesture commands to execute operations. Gesture commands can be defined through the gesture management model. Users can input a desired gesture into this system. After training these gesture commands by a neural network, users can freely use them to replace menus. The following figure shows an undo gesture command.

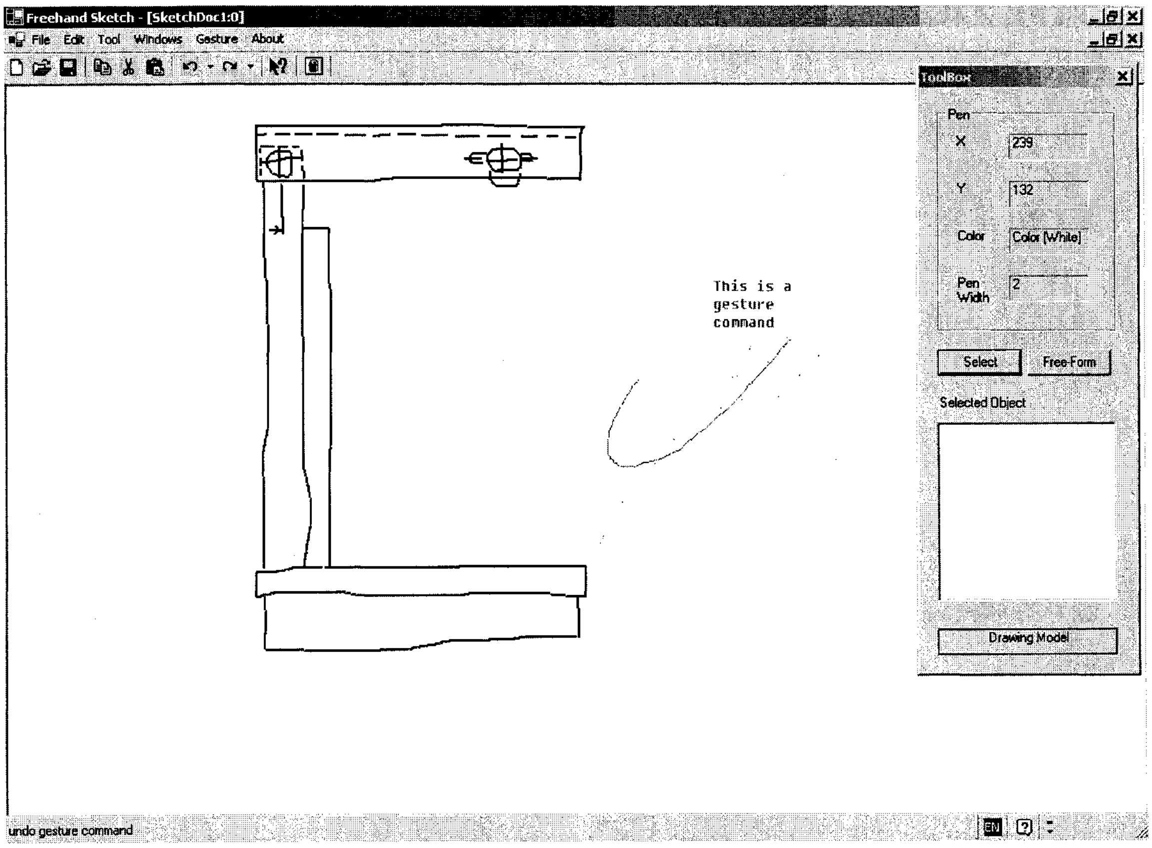


Figure 19 Before Undo Gesture Command

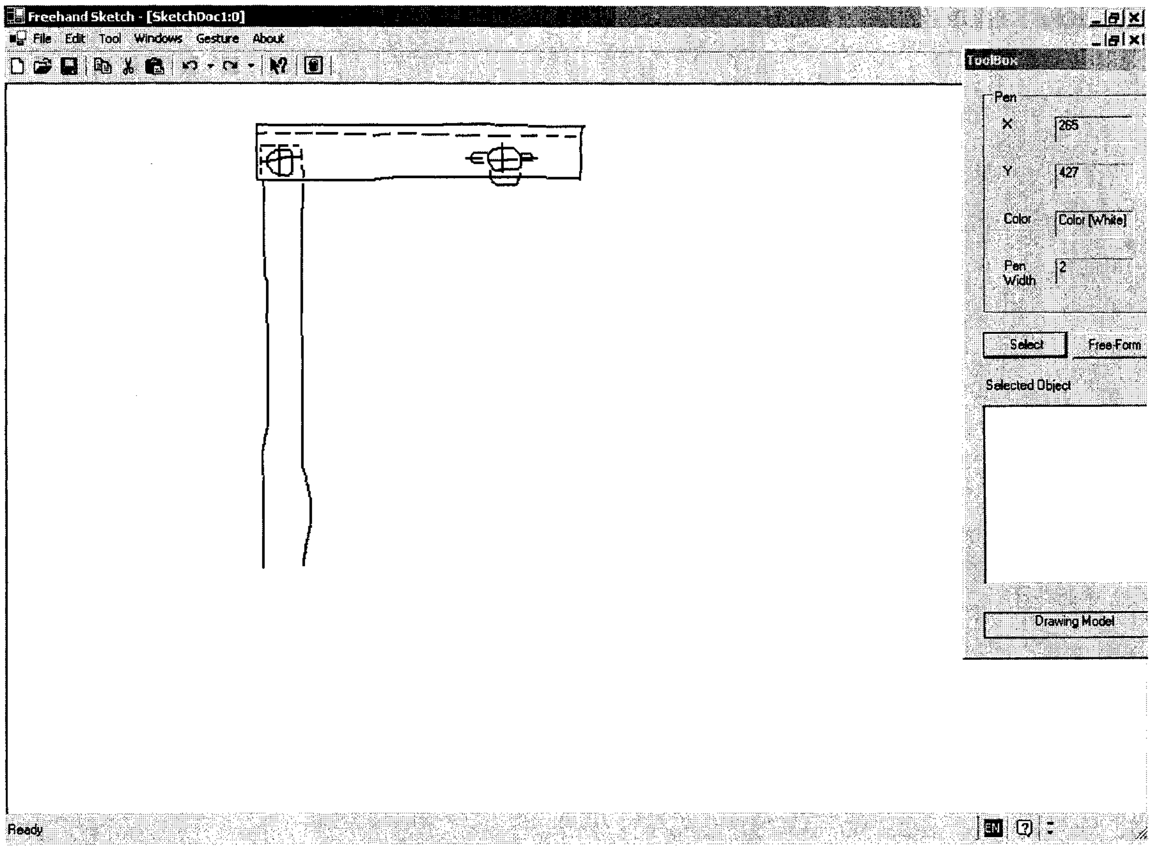


Figure 20 After Several Undo Gesture Commands

The following figure shows the final sketch that is created by our system.

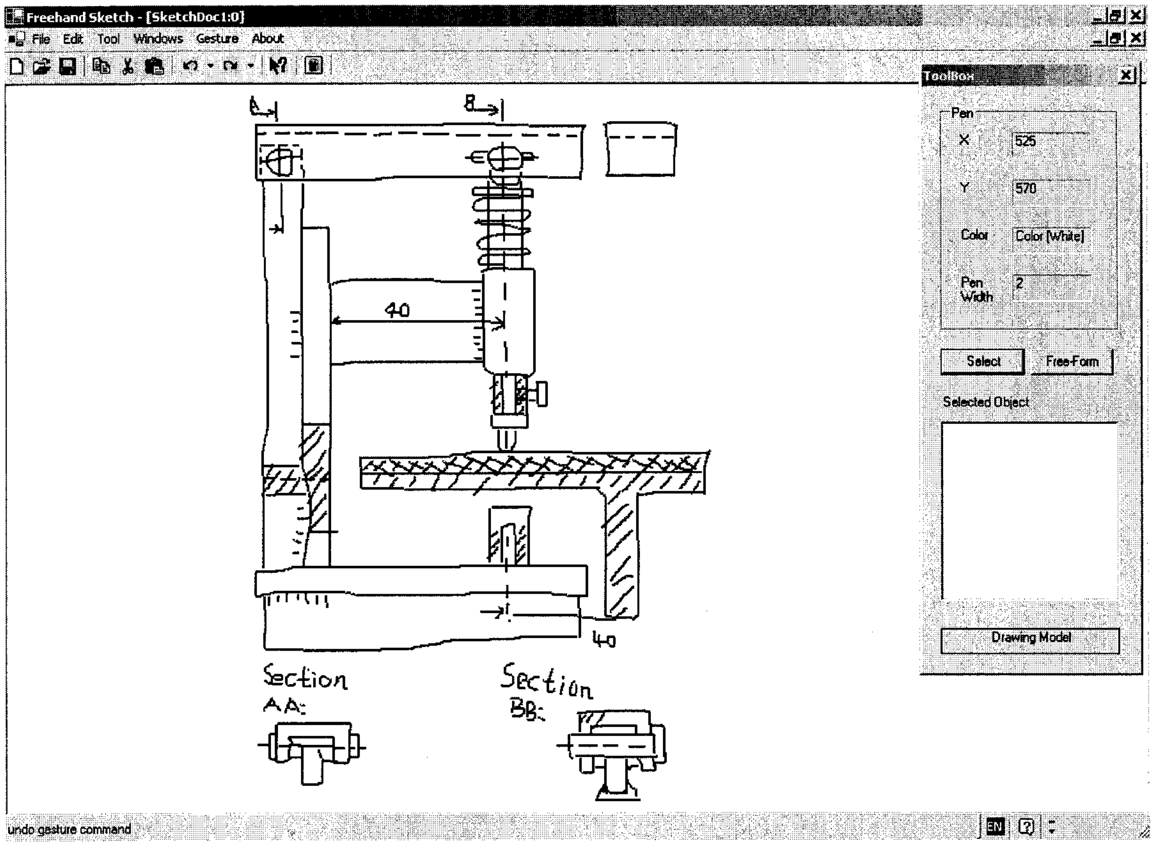


Figure 21 Final Sketch

5.4.2 Merge and Break

As we know, each design sketch can be divided into three parts: gesture, text and geometry (Zeng, Pardasani et al. 2004). Designers can select the desired text strokes and merge them into one text object. This merging operation is recursive. The final result could be a sketch object, which would include three different objects: gesture object, text object and geometry object. The following figures present how to use merge operations to group different objects. First, merging geometry strokes into geometry objects; second, merging text strokes into text objects; third, merging these two objects into a single object. In this case, there is no gesture object in the final sketch object. After that, one can execute any operation on this object, such as the deleting operation.

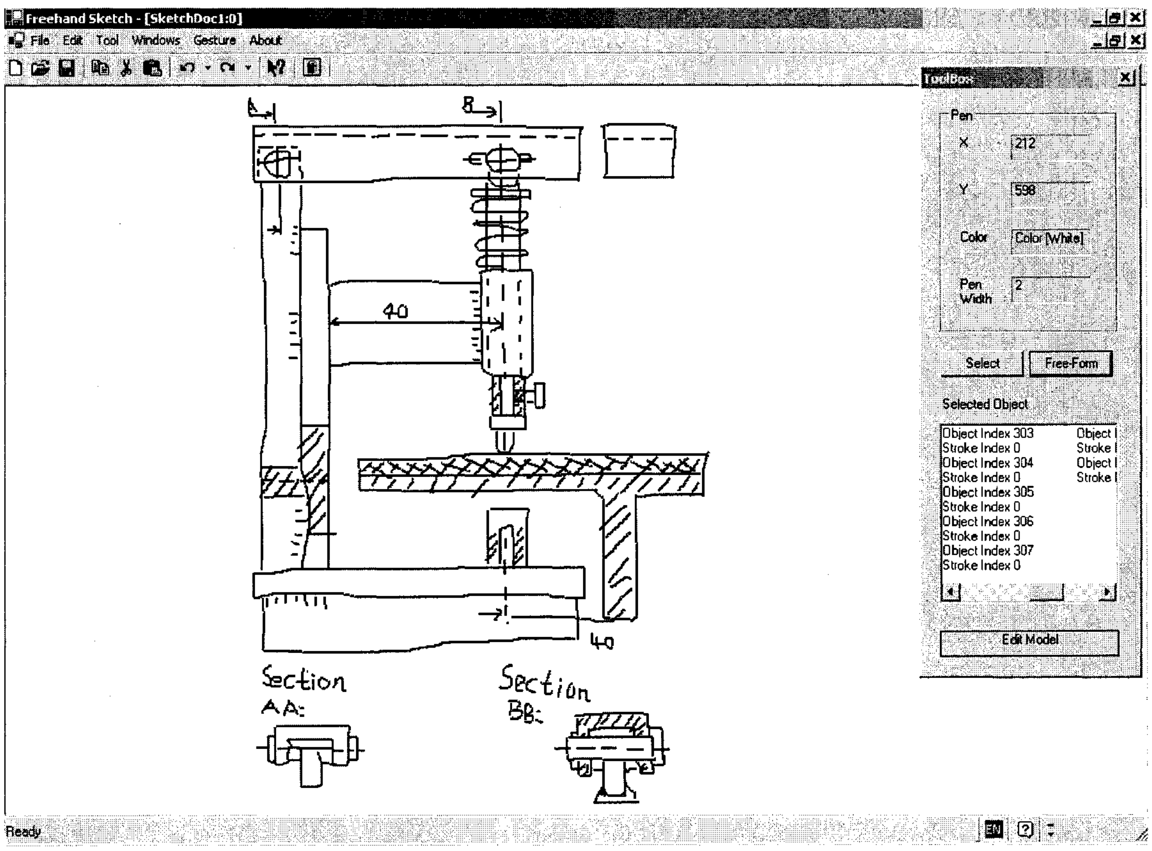


Figure 22 Merge and Create a Geometry Object

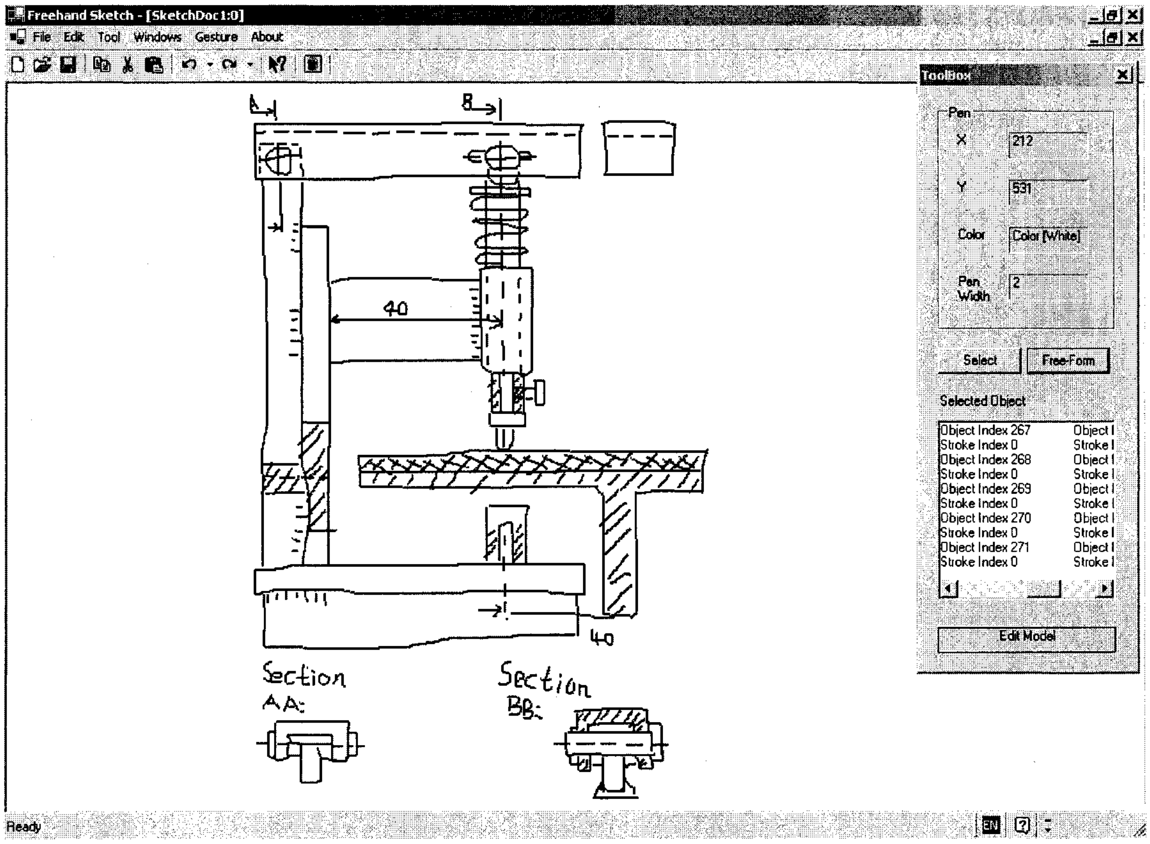


Figure 23 Create a Text Object and Merge Two Objects

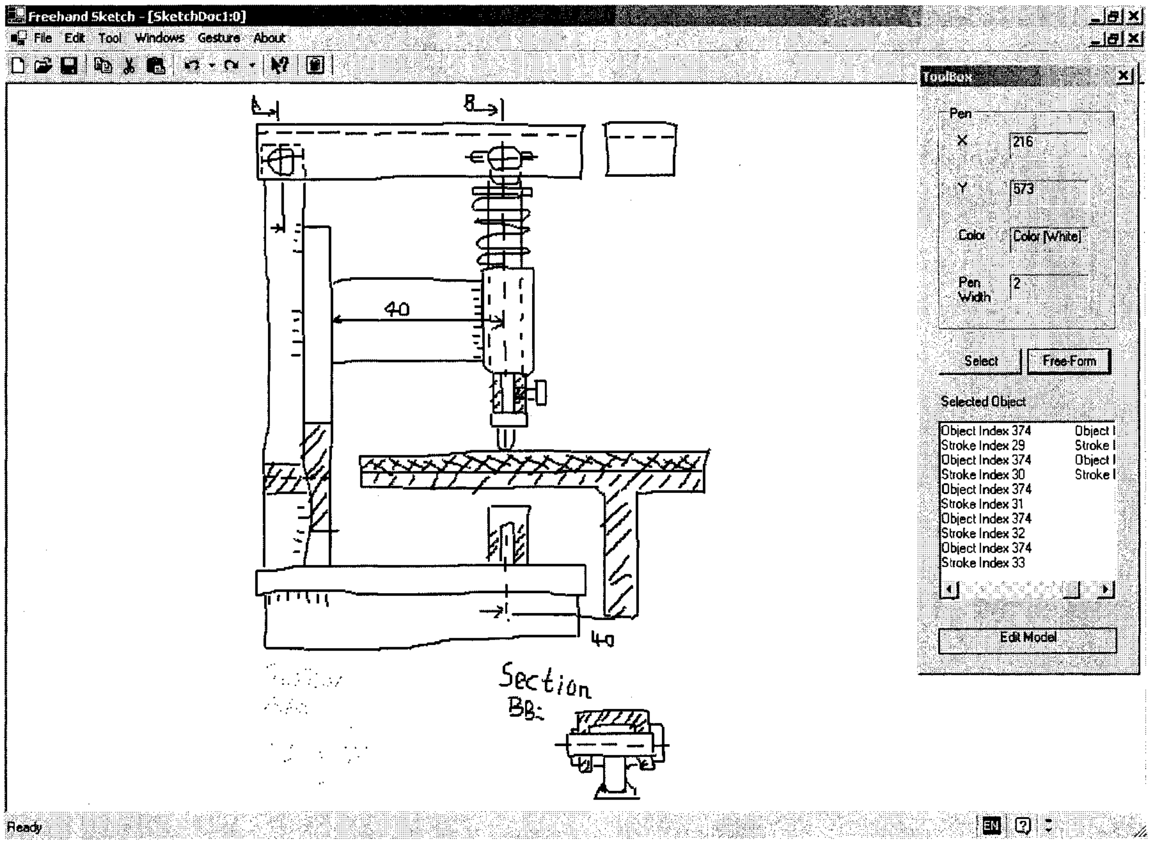


Figure 24 Delete the Final Object

Break operations can break one object into several low-level sub-objects. The following examples show how to break the object above into a text object and a geometry object and then how to combine the geometry object with another object and delete the result.

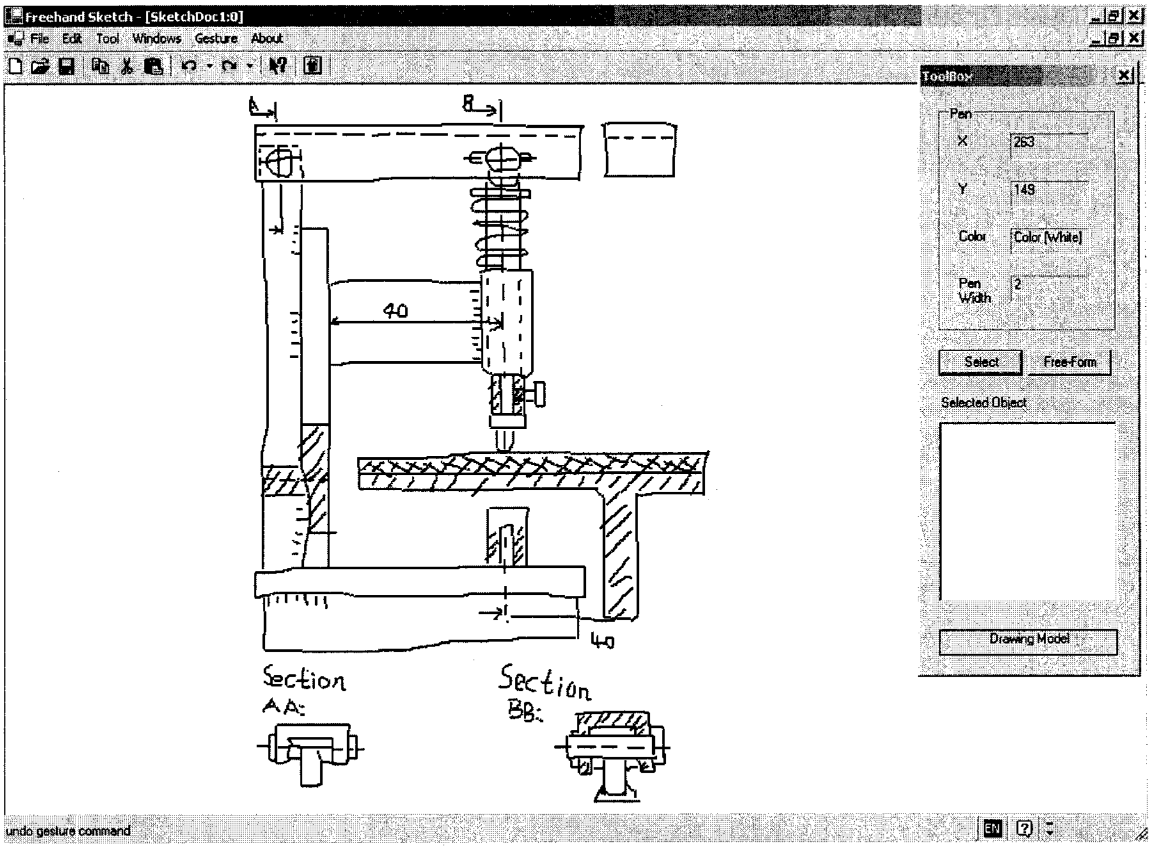


Figure 25 Break the Final Object

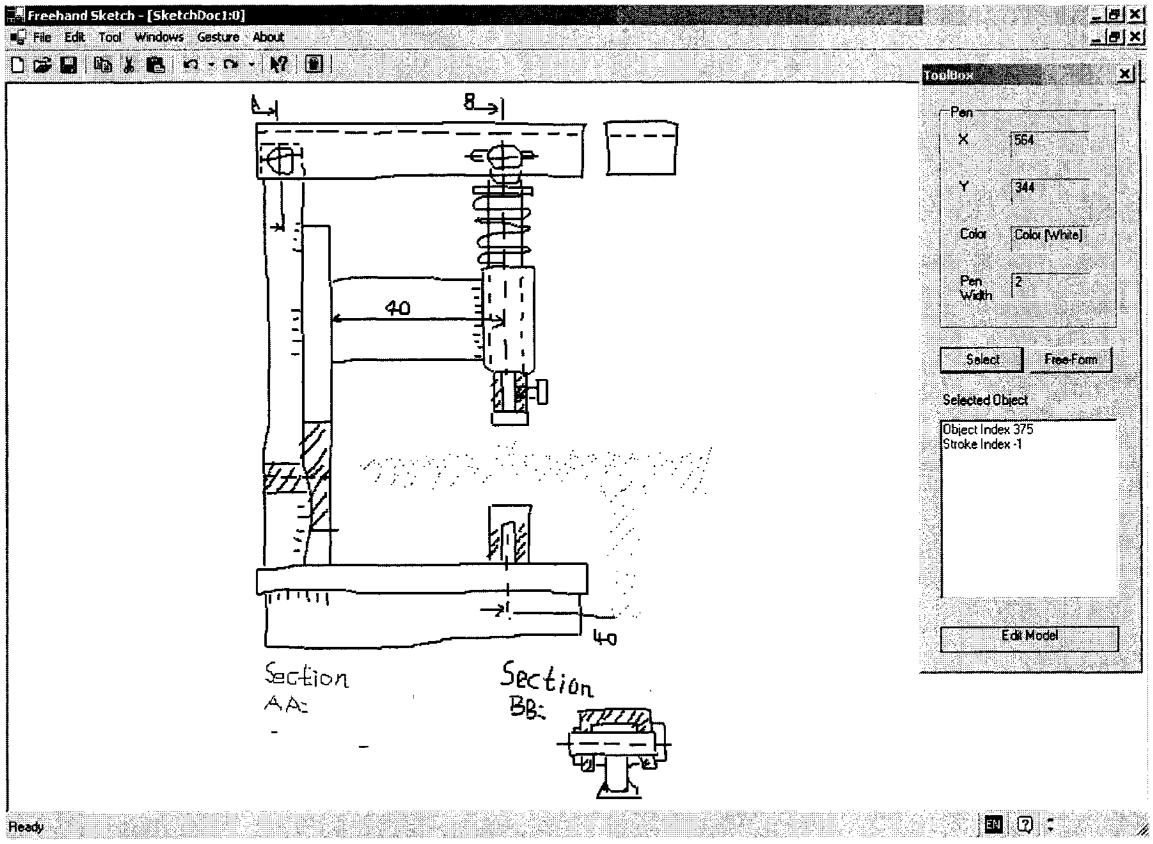


Figure 26 Delete the Geometry Object with Another Object

After designers complete the merging operation, we can apply the design of algorithms to define the design sketch well. This is our first step towards providing designers with a natural interface.

Chapter 6

Future Work

There are three types of future research that must be continued. The first is to integrate a sketch recognition component into this project. Although some technologies related to sketch recognition are discussed in this thesis, they are not powerful enough to implement sketch recognition. Recognizing a sketch drawn freely is not an easy task. Many researchers have already developed some systems to recognize a rough sketch and to replace it by a formal sketch. However, these systems cannot be applied in a real world because the efficiency and accuracy of these systems are inadequate.

Secondly, future research will involve designing a proper experiment to measure the mental workload of designers when they complete design tasks with our sketching system. A compatible experiment could help us find the shortage of this system as well as to validate our research results. Because researchers suggest that the actions of people's eyes have a strong relation with people's mental workload, a rough idea of this experiment would consist of using an eye tracking equipment to design a good system to record the actions of a designer's eyes. Two sketching systems will be used in this experiment. One is a traditional system like a CAD system with menus and buttons. The other is our sketching system with gesture commands. After analyzing these data about a designer's eye's actions, we can easily compare both systems and come to a conclusion.

Last, this system is actually applied to a big ongoing system that attempts to provide industry designers with a powerful computer-aided tool. This project is based on Dr. Zeng's design theory: an environment-based design. The sketching system will be integrated with other components, such as an analysis system of user requirements. These systems would work together harmoniously to provide industry designers from collecting user requirements to creating some rough products.

Chapter 7
Conclusion

Conceptual designers reject existing computer-aided design tools because there is a considerable difference between the user's mental models and the actual mental model of the designers using the systems. This present thesis develops a designer's mental model by applying the axiomatic theory of design modeling. Based on this mental model, a natural sketching interface is designed to support the designer's conceptual design process.

A major contribution of this thesis is the identification of the gap between the user's preliminary mental model and the user mental model underlying the designed product, based on which a useful guideline for UI designers is proposed.

Bibliography

Allan, J. K. and Mistree, F. (1997) "Decision-based design: where should we go from here?" from http://dbd.eng.buffalo.edu/papers/dbd-where.html

Altshuller, G. S. (1984). *Creativity as an Exact Science: the Theory of the Solutions of Inventive Problems*, Gordon and Breach Science Publishers.

Braha, D. and Maimon, O. (1998). *A Mathematical Theory of Design*, Kluwer Academic Publishers.

Buchal, R. O. (2002). "Sketching and Computer-Aided Conceptual Design". *Proceedings of the International Conference on Computer Supported Cooperative Work in Design*, Rio de Janeiro.

Carroll, J. M. and Olson, J. (1988). Mental models in human-computer interaction. *Handbook of Human-computer Interface*. Helander, M. Amsterdam, Elsevier.

Craik, K. (1943). *The Nature of Explanation*. Cambridge, Cambridge University Press.

Cross, N. (1999). "Natural intelligence in design." Design Studies **20**(1): 25-39.

di Sessa, A. A. (1986). Models of computation. *User Centered System Design: New Perspectives on Human-Computer Interaction*. Norman, D. A. and Draper, S. W. Hillsdale, NJ, Erlbaum: 201-218.

Eekels, J. (2000). "On the fundamentals of engineering design science: the geography of engineering design science. Part 1." Journal of Engineering design **11**(4): 377-397.

Ferguson, E. S. (1992). *Engineering and the Mind's Eye*. Cambridge MA, MIT Press.

Figen, U., Andrew, F. and Norio, A. (1996). "On-line shape recognition with incremental training using binary synaptic weights algorithm." Applied Intelligence **6**(3): 225-240

Gentner, D. and Stevens, A. (1983). *Mental Models*. Hillsdale, NJ, Erlbaum.

Gero, J. (2000). *Artificial Intelligence in Design '00.* Norwell, MA, USA, Kluwer Academic.

Goel, V. (1992). "Comparison of well-structured & ill-structured task environments and problem spaces". *Proceedings of The Fourteenth Annual Conference of the Cognitive Science Society* Hillsdale, NJ, Erbaum.

Goel, V. (1992). "Ill structured representations for ill-structured problems". *Proceedings of the fourteenth annual conference of the cognitive science society* Hillsdale, NJ, Erbaum.

Goel, V. (1995). *Sketches of Thought.* Cambridge MA, MIT Press.

Goldschmidt, G. (1991). "The dialectics of sketching." Creativity Research Journal **4**(2): 123-143.

Gross, M. D. (1996). "The Electronic Cocktail Napkin--a computational environment for working with design diagrams." Design Studies **17**(1): 53-69.

Gross, M. D., Ervin, S. M., Anderson, J. A. and Fleischer, A. (1988). "Constraints: knowledge representation in design." Design Studies **9**(3): 133-143.

Ho, C.-H. (2001). "Some phenomena of problem decomposition strategy for design thinking: differences between novices and experts." Design Studies **22**(1): 27-45.

Hubka, V. (1984). "Principles of Engineering Design". *Butterworth Scientific.*, London.

Hubka, V. and Eder, W. (1988). *Theory of Technical Systems*, Springer-Verlag.

John, B. E. (1998). "Cognitive modeling for human-computer interaction". *Proceedings of Graphics Interface '98*, Vancouver, British Columbia, Canada

Kavakli, M., Scrivener, S. A. R. and Ball, L. J. (1998). "Structure in idea sketching behaviour." Design Studies **19**(4): 485-517.

Knight, K. (1999). *The Catholic Encyclopedia*, XI, (http://www.newadvent.org/cathen//11090c.htm), Online Edition.

Landay, J. A. and Myers, B. A. (2001). "Sketching interfaces: toward more human interface design." IEEE Computer **34**(3): 56-64.

Liu, W., Cha, J. Z., Xu, X. H. and Guo, W. J. (2002). "Online hand-sketched engineering drawing neural network recognition". *TENCON '02. Proceedings. 2002 IEEE Region 10 Conference on Computers, Communications, Control and Power Engineering.*

Martina, A. S. (1997). "Eliciting and describing users' models of computer systems". Computer Science, University of Birmingham. **Ph.D.**

Newell, A. and Simon, H. (1972). *Human Problem Solving*. Englewood Cliffs, NJ, Prentice-Hall.

Nielsen, J. (1990). "A meta-model for interacting with computers." Interacting with Computers **2**(2): 147-160.

Norman, D. A. (1988). *The Design of Everyday Things*. New York, Doubleday Press.

Pahl, G. and Beitz, W. (1988). *Engineering Design: A Systematic Approach*, Spinger-Verlag.

Pahl, G. and Beitz, W. (1997). "Konstruktionslehre". *Springer*, Berlin.

Plimmer, B. and Apperley, M. (2002). "Computer-aided sketching to capture preliminary design". *The Third Australasian User Interfaces Conference (AUIC2002)*, Melbourne, Australia.

Purcell, A. T. and Gero, J. S. (1998). "Drawings and the design process: A review of protocol studies in design and other disciplines and related research in cognitive psychology." Design Studies **19**(4): 389-430.

Qin, S.-F., Wright, D. K. and Jordanov, I. N. (2000). "From on-line sketching to 2D and 3D geometry: A system based on fuzzy knowledge." CAD Computer Aided Design **32**(14): 851-866.

Qin, S.-F., Wright, D. K. and Jordanov, I. N. (2001). "Conceptual design tool: A sketch and fuzzy logic based system." ImechE, Proc Instn Mech Engrs, Part B: Journal of Engineering Manufacture **215**: 111-116.

Qin, S.-F., Wright, D. K. and Jordanov, I. N. (2001). "On-line segmentation of freehand sketches by knowledge-based nonlinear thresholding operations." Pattern Recognition **34**(10): 1885-1893.

Rauterberg, M. (1995). "From novice to expert decision behaviour: a qualitative modelling approach with petri nets". *Simbiosis of Human and Artifact*, Anzai Y.

Redish, E. (1994). "The implication of cognitive studies for teaching physics." The American Journal of Physics **62**(6): 796-803.

Rittel, H. W. J. (1972). "Second-generation design methods". The DMG 5th Anniversary Report, DMG Occasional Paper No. **1**: 5-10.

Romer, A., Leinert, S. and Sachse, P. (2000). "External support of problem analysis in design problem solving." Research in Engineering Design **12**(3): 144-151.

Rowe, P. G. (1987). *Design Thinking*, MIT Press.

Saja, A. D. (1985). "The cognitive model: an approach to designing the human-computer interface." ACM SIGCHI Bulletin **16**(3): 36-40.

Salustri, F. A. and Venter, R. D. (1992). "An axiomatic theory of engineering design information." Engineering with Computers **8**(4): 197-211.

Schon, D. A. and Wiggins, G. (1992). "Kinds of seeing and their functions in designing." Design Studies **13**(2): 135-156.

Sifaqui, C. (1999). "Structuring user interfaces with a meta-model of mental models." Computers and Graphics **23**(3): p323-330.

Simon, H. A. (1969). *The Sciences of the Artificial*, MIT Press.

Staggers, N. and Norcio, A. F. (1993). "Mental models: concepts for human-computer interaction research." International Journal of Man-Machine Studies **38**: 587-605.

Suh, N. (1990). *Principles of Design*. New York, Oxford University Press.

Suzuki, Y., Itakura, K.-i., Saga, S. and Maeda, J. (2001). "Signal processing and pattern recognition with soft computing". *Proceedings of the IEEE.*

Tang, H.-H. and Gero, J. S. (2001). Sketches as affordances of meanings in the design process. *Visual and Spatial Reasoning in Design II.* Gero, J. S., Tversky, B. and Purcell, T. Key Centre of Design Computing and Cognition, University of Sydney, Australia: 271-282.

Tomiyama, T. and Yoshikawa, H. (1987). Extended general design theory. *Design Theory for CAD, Proceedings IFIP WG5.2 Working Conference on Design Theory for CAD.* Yoshikawa, H. and Warman, E. A. Amesterdam, Elsevier Science Publishers B. V.: 95-130.

Tovery, M. (1997). "Styling and design: intuition and analysis in industrial design." Design Studies **18**: 5-13.

Tovey, M. (1997). "Styling and design: intuition and analysis in industrial design." Design Studies **18**(1): 5-31.

Tovey, M., Porter, S. and Newman, R. (2003). "Sketching, concept development and automotive design." Design Studies **24**(2): 135-153.

Turk, M. and Robertson, G. (2000). "Perceptual user interfaces." Communications of the ACM **43**(3): 32-70.

Ullman, D. G. (2002). *The Mechanical Design Process*, McGraw Hill.

Van Dalen, D., Doets, H. C. and De Swart, H. (1978). *Sets: Naive, Axiomatic, and Applied*, Pergamon Press.

van Dijk, C. G. C. (1995). "New insights in computer-aided conceptual design." Design Studies **16**(1): 62-80.

Verstijnen, I. M., van Leeuwen, C., Goldschmidt, G., Hamel, R. and Hennessey, J. M. (1998). "Sketching and creative discovery." Design Studies **19**(4): 519-546.

Vijay, I. (2005). "http://cnmat.cnmat.berkeley.edu/People/Vijay/02.%20Defining%20terms.html." from http://cnmat.cnmat.berkeley.edu/People/Vijay/02.%20Defining%20terms.html.

Weinstein, S. and Graves, R. E. (2002). "Are Creativity and Schizotypy Products of a Right Hemisphere Bias?*1." Brain and Cognition **49**(1): 138-151.

Yoshikawa, H. (1981). General design theory and CAD system. *Man-Machine Communication in CAD/CAM, Proceedings IFIP WG 5.2 Working Conference*. Sata, T. and Warman, E. A. Amsterdam, North Holland: 35-38.

Zeng, Y. (2002). "Axiomatic theory of design modeling." Transaction of SDPS: Journal of Integrated Design and Process Science **6**(3): 1-28.

Zeng, Y. and Cheng, G. D. (1991). "On the logic of design." Design Studies **12**(3): 137-141.

Zeng, Y., Pardasani, A., Antunes, H., Li, Z., Dickinson, J., Gupta, V. and Baulier, D. (2004). "Mathematical foundation for modeling conceptual design sketches." Transactions of the ASME: Journal of Computing and Information Science in Engineering **4**(2): 150-159.